časopis za popularizaciju informatike i računarstva

računari

u vašoj školi/za početnike

tematsko izdanje
za škole i početnike

izdaje bigz
YUISSN 032552-7271
cena 1 500 din.

kako radi računar

sve periferijske jedinice

sve o pc računarima

algoritmi: zbirka rešenih zadataka

100 strana

samogradnja tima 011

kompletna škola bejzika





Uz naslovnu stranu

Da bi nastava informatike u školama oživela potrebne su dve stvari dobra [illegible] dobar nastavnički kadar Literatur[illegible] redakcija Računara a [illegible]stavu drže Eliza Pandališevska i Maša P[illegible]ć Fotografija je nastala na Višoj školi z[illegible]konomiku i informatiku u organiza[illegible] Bratislava Đorđevi[illegible] realizaciji Darka Todića

Izdaje Beogradski izdavačko-grafički zavod 11000 Beograd Bulevar vojvode Mišića 17
Generalni direktor Dobrosav Petrović
Glavni i odgovorni urednik časopisa „Galaksija" Stanko Stojiljković
Glavni i odgovorni urednik Jovan Regasek
Stručni urednik izdanja Nevenka Spalević
Redaktor izdanja Žarko Vukosavljević
Likovno-grafički urednik Dušan Mijatović **Tehnički urednik** Mirko Popov
Stručna redakcija Žarko Berberski, Voja Gašić, Slobodan Perović, Dejan Ristanović, Jovan Skuljan, prof dr Dušan Slavić, Nevenka Spalević, Zoran Životić, Anđelko Zgorelec
Sekretar redakcije Zorka Simović **Marketing** Sergije Marčenko

Adresa redakcije
11000 Beograd, Bulevar vojvode Mišića 17/III
Telefoni 650-161 (sekretarijat) 653-748 (redakcija) 650-528 (prodaja) 651-793 (propaganda)

Rukopisi se ne vraćaju
Štampa Beogradski izdavačko-grafički zavod 11000 Beograd Bulevar vojvode Mišića 17
Na osnovu mišljenja Republičkog sekretarijata za kulturu broj 413-77/72-05 i „Službenog glasnika" broj 26/72, ovo izdanje oslobođeno je poreza na promet

računari

tematsko izdanje za škole i početnike
april 1988. cena 1500 din

Sadržaj

OSNOV

Kako, kada i zašto su računari uvođeni u naše škola? Kakva je koncapcija informatičkog i računarskog obrazovanja u Srbiji? Na koji su način učenici osnovnih škola pobili mišljenja nakih pedagoških stručnjaka da nisu dorasli učenju informatike i računarstva? Kako škola (ne)korista računarsku opramu? Na ta i druga pitanja odgovor iz prva ruke pruža dipl. inž. Dragan Vasić, republički savetnik za informatiku koji je bio zadužen za uvođenje računara u škola u Srbiji.

Prvi koraci na uvođenju računara u škole u SR Srbiji započeti su 1974 godine kada je jedna grupa stručnjaka okupljenih oko Laboratorije za sisteme i informatiku Instituta Boris Kidrič u Vinči pokušala da napravi prve projekcije mogućnosti korišćenja računara u nastavi i učenju Kao rezultat ovih napora decembra 1974 godine organizovano je savetovanje Nastava i učenje uz pomoć računara Tada je prvi put u Republici obavljen teleprocessing obrazovnog softvera Učesnici savetovanja kojih je bilo oko 300 imali su priliku da komuniciraju sa računskim centrom u Vinči i da se prvi put susretnu sa obrazovnim softverom Kao terminal poslužio je običan teleks uređaj!

Prvi računari ušli su u škole u SR Srbiji krajem sedamdesetih i početkom osamdesetih godina Uveli su ih retki entuzijasti koji su u pravom trenutku shvatili ulogu i značaj računara u obrazovanju Tako se računari javljaju u Elektrotehničkoj školi Nikola Tesla u Beogradu valjevskoj Gimnaziji Ekonomskoj srednjoj školi u Valjevu Elektrotehničkom školskom centru u Nišu Matematičkoj gimnaziji u Beogradu ali i u nekim seoskim sredinama, kao što je na primer Osnovne škola u Lozoviku Tipovi računara bili su različiti Najviše je bilo računara epl II a u Nišu je bio instaliran jedan sistem hanivel sa tri terminala koja su radila na principu time-sharing sistema.

Novim nastavnim planom 1977/78 školske godine u neke struke uvedeni su informatički predmeti U okviru Osnova tehnike i proizvodnje uvedeno je 12 časova informatike i računarstva kao priprema za računarsku pismenost Ovi sadržaji nisu zadovoljavali interesovanje učenika jer su se gotovo po pravilu realizovali kao teorijska nastava bez praktičnog rada

Od 1980 do 1985 godine Centar za multidisciplinarne studije Univerziteta u Beogradu i Institut Mihajlo Pupin radili su projekt Računar u obrazovanju U drugu fazu ovog projekta uključen je Čezli koledž (Univerziteta u Londonu) Cilj ovog projekta bio je projektovanje mikroračunarskog sistema za potrebe škola usmerenog obrazovanja zasnovanog pretežno na domaćim komponentama kao i razvoj domaćeg programskog sistema U okviru projekta izrađen je prototip računara tim 10 čije su grafičke mogućnosti prilagođene potrebama obrazovanja Projekt nije do kraja realizovan jer nije obavljen transfer obrazovnih softverskih paketa iz Velike Britanije.

Otpori i nerazumevanje

Republički zavod za unapređivanje vaspitanja i obrazovanja tokom 1983 godine počinje intenzivno da



Dragan Vasić

PROTIV STRUČNJAKA

radi na uvođenju informatike i računarstva u vaspitno-obrazovni rad. U to vreme bilo je dosta otpora i nerazumevanja. Mnogi su osporavali da učenici osnovne škole mogu da savladaju te sadržaje, pa je izrađen projekt za eksperiment „Uvođenje nastave informatike i računarstva u programe obrazovno-vaspitnog rada osnovnog obrazovanja". Ovaj eksperiment usvojio je Prosvetni savet SR Srbije pa je sproveden školske 1984/85. godine. U ogledu je učestvovalo 376 učenika iz 12 osnovnih škola.

Rezultati su prijatno iznenadili: 93,83% učenika osmog razreda osnovnih škola sa uspehom je savladalo ponuđeni model nastave informatike i računarstva. Prosečna ocena na testu znanja za čitav uzorak bila je 3,39, za 0,22 veća od prosečne ocene za sve ostale predmete.

Utvrđeni su značajni sredinski uticaji na rezultate testa znanja. Učenici iz prigradskih škola postigli su prosečnu ocenu 2,31, učenici sa sela 3,47, a učenici iz grada 4,34.

Nastavu informatike i računarstva smatralo je interesantnom 85,27% učenika, 79,06% izrazilo je želju da nastavi ovo izučavanje, a 70,54% učenika smatralo je da im je ova nastava korisna za njihovu buduću struku.

Posle ovog eksperimenta, otpor u nekim obrazovnim krugovima su bitno smanjeni, mada ne i potpuno. Rezultati istraživanja su ugrađeni u strategiju uvođenja informatike i računarstva na teritoriji SR Srbije bez pokrajina.

Korak po korak

Kao datum kad je počelo sistematsko uvođenje informatike i računarstva u vaspitno-obrazovni rad na teritoriji SR Srbije bez pokrajina treba uzeti 20. decembar 1985, kada je Izvršno veće Skupštine SR Srbije usvojio „Analizu potreba i mogućnosti šireg uvođenja i korišćenja računara u procesu obrazovanja", koju je izradio Republički zavod za unapređivanje vaspitanja i obrazovanja.

Prema tom dokumentu, osnova politike uvođenja informatike i računarstva treba da se zasniva na:

- Potrebi šireg uvođenja nove informacione tehnologije u sve oblasti rada, s obzirom na to da su informatika i računarstvo osnova treće tehnološke revolucije.
- Planskom i sistematskom obrazovanju nastavnika za primenu računara u vaspitno-obrazovnom radu, i to kako u okviru izučavanja informatike i računarstva kao nastavne discipline, tako i u primeni računara u svim nastavnim predmetima.
- Izradi obrazovnog softvera, kome treba posvetiti istu (ako ne i veću) pažnju kao i opremanju škola računarima.
- Planskom opremanju škola standardizovanim računarima.
- Potrebi korišćenja računara u svim nivoima obrazovanja.

U to vreme, krajem 1985. godine, u osnovnim i srednjim školama na teritoriji SR Srbije bez pokrajina bio je instaliran ukupno 441 računar.

Većina predviđenih poslova je obavljena. Nisu obezbeđena sredstva za proizvodnju obrazovnog softvera, mada su predviđena društvenim planom. Sistematska proizvodnja obrazovnog softvera nije započeta, a nije ni postignut dogovor proizvođača oko specijalizacije u proizvodnji računara za potrebe obrazovanja.

Korisno odugovlačenje

Kriterijumi za izbor računara za potrebe obrazovanja jedinstveni su za čitavu teritoriju SR Srbije i usaglašeni su 1985. i 1986. godine.

Osnovni kriterijumi su:

a) Računar se može zasnivati na osmobitnom procesoru.

b) Kapacitet memorije treba da bude najmanje 64KB.

c) Računar mora raspolagati grafikom srednje rezolucije od najmanje 256×192 tačke sa programskom kontrolom na nivou svake tačke, uz mogućnost istovremenog prikazivanja teksta i grafike, kao i prikazivanja teksta inverzno, sa svetlucanjem i u poluintenzitetu.

d) Poželjno je da računar ima najmanje četiri boje.

e) Mogućnost programiranja zvuka na tri kanala u opsegu do najmanje tri oktave.

f) Tastatura treba da ima standardni set srpskohrvatske azbuke sa mogućnošću prikazivanja malih i velikih slova latinice i ćirilice.

g) Računar treba da ima priključke za kasetofon, disk-jedinicu i štampač, AD/DA konvertor radi priključivanja na laboratorijsku opremu, kao i priključak za spoljašnji EPROM.

Nedostatak ovih kriterijuma je u tome što nije precizirano ništa u vezi sa aplikativnim softverom.

Posle više sastanaka različitih komisija i dosta odugovlačenja, aprila 1987. donesena je odluka o tipovima računara koji se mogu koristiti u osnovnim i srednjim školama na teritoriji SR Srbije bez pokrajina. Izabrani su računari za koje se elaboratom Instituta za računarsku tehniku Instituta „Boris Kidrič" potvrđuje stoprocentna kompatibilnost sa IBM PC XT računarima. Kao radne stanice, uz ove računare mogu biti korišćeni u mreži:

a) „pekom 64" (EI Niš)
b) „tim 10" (Instituta „Mihajlo Pupin", Beograd)
c) „partner I FG/Š" („Iskre Delte", Ljubljana)

Odugovlačenje sa donošenjem odluke, pored nedostataka, imalo je i prednosti. Nedostatak je što opremanje škola nije bilo moguće započeti ranije, a prednost je što je izborom PC mašina postignut standard veći od zahtevanog u kriterijumima. Naime, gotovo je sigurno da je izbor obavljen ranije, da PC kompatibilan računar ne bi bio izabran.

Od poletarca od srednjoškolca

Na teritoriji SR Srbije bez pokrajina radi 948 predškolskih vaspitnih organizacija. Na upitnik o stanju informacione tehnologije odgovorilo je 86.

Nijedna od njih nema ni kućni ni personalni računar.

Od 887 matičnih osnovnih škola upitnik o opremljenosti vratile su 364 škole, koje su u decembru 1987. imale 324 kućna i 58 personalnih računara.

Najbrojniji kućni računar je „Galaksija 6—6" (120). Potom slede „lola 8" (84), „spektrum 48" (43), „spektrum 16" (20), „pekom" i jedan „komodor 64". Od ličnih računara najzastupljeniji su „ulira" (16), „komodor PC 20" (12) i „komodor PC 10" (3).

Treba reći da u osnovnim školama čitave Jugoslavije ima ukupno 155 personalnih računara.

Od 253 školska centra srednjeg usmerenog obrazovanja na anketu je odgovorilo 85.

U njima je bilo 223 kućna i 20 personalnih računara. Najviše je „komodora 64" (47). Zatim slede „lola 8" (35), „orik" (34), „galaksija" (22), „spektrum 16K" (20), „spektrum 48K" (20), „pekom" (19), „oreo" (4) i „QL" (1) i „šnajder" (1). Osim toga, u školama ima i 20 kućnih računara drugih tipova.

Među ličnim računarima su „epl IIe" (6), „epl II c" (2), „mekintoš" (1) i 8 računara za koje se iz upitnika ne može utvrditi koje su marke.

Podatke o smeštaju informatičke opreme dostavilo je 78 srednjih škola. Računarska oprema je smeštena u specijalizovani kabinete u 42 srednje škole. Dvadeset računara je smešteno u nespecijalizovane učionice. Samo u jednoj srednjoj školi računar je bio u medijateci. Za 20 računara nisu dostavljeni podaci o tome gde su smešteni. Iskustvo iz obilaska terena govori da se računari ponekad mogu naći zaključani u sobi direktora škole.

Od 135 osnovnih i 26 srednjih škola za decu i omladinu sa teškoćama u razvoju na upitnik je odgovorilo svega 20 osnovnih i 6 srednjih.

U ovoj oblasti se moraju učiniti posebni napori radi popravljanja stanja. Primeri iz Velike Britanije i nekih drugih zemalja govore da su, na primer, primenom računara deca sa teškoćama u mentalnom razvoju naučila da se oblače u toku od tri meseca, što u klasičnoj nastavi nije uspelo za dve godine.

Prve laste

U upitnik je uključeno i pitanje o broju nastavnika koji je osposobljen za rad sa računarima, to jest o nastavnicima koji su se u toku studija upoznali sa rukovanjem računara, što naravno ne znači da su upoznati sa didaktičko-metodičkim principima primene novih informacionih tehnologija. Ti nastavnici su prve laste nove ere u obrazovanju i vaspitavanju.

U 86 predškolskih organizacija koje su odgovorile na upitnik radilo je 2486 vaspitača. Od tog broja za rad sa računarima je osposobljen 31 vaspitač, odnosno 1,2%. Niko od njih nije koristio računare u vaspitno-obrazovnom radu.

U 364 osnovne škole radilo je 13069 nastavnika, od kojih je 259 obučeno za primenu računara u obrazovanju, što čini 1,98% od nastavnika obuhvaćenih upitnikom. Od njih 193 koristi računare u vaspitno-obrazovnom, odnosno 1,43% od ukupnog broja nastavnika.

Prema podacima prikupljenim upitnikom u 85 centara srednjeg usmerenog obrazovanja radi 3729 nastavnika. Od njih je 215 ili 5,75% obučeno, odnosno ima odgovarajuću spremu za rad sa računarima.

Nastavu informatike i računarstva drži 50 redovnih nastavnika i 73 nastavnika iz udruženog rada. U vannastavnim aktivnostima računare koriste 132 nastavnika. O računarima brine 85 nastavnika, dakle u svakom srednjoškolskom centru jedan nastavnik je zadužen za računare.

A programi?

Aplikacioni softver namenjen obrazovanju gotovo da ne postoji. Zavod za udžbenike i nastavna sredstva je izdao oko 150 programa za računare „galaksija", „spektrum" i „komodor". Međutim, prema podacima prikupljenim iz upitnika, aplikacioni programi se gotovo ne koriste. To ukazuje na nizak nivo računarskog obrazovanja u nas, jer je danas nezamislivo imati računar a ne koristiti programe za procesiranje teksta, za obradu tabela ili baza podataka. Očito preovlađuje uverenje da je dovoljno nabaviti računare i onda ih koristiti za učenje različitih programskih jezika i programiranje, što je samo jedan vid korišćenja računara.

Gotovi programi su se, sem izuzetka u jednoj osnovnoj školi, koristili isključivo u centrima srednjeg usmerenog obrazovanja. U nastavi informatike i računarstva koristilo se prosečno 5 programa po 6,5 sati godišnje. U nastavi prirodnih nauka korišćen je bio prosečno jedan program jedan sat godišnje. Za društveno-jezičko područje nije bilo programa, kao ni za ostale opšte predmete.

U slobodnim aktivnostima korišćen je prosečno 1,5 program po 2,5 sata godišnje.

Od 86 predškolskih ustanova, 364 osnovne škole i 85 srednjih škola koje su odgovorile na upitnik, nijedna ne koristi računare za upis učenika, pedagošku evidenciju, evidenciju nastavnih časova, raspored časova, bibliotečko-medijatečku evidenciju, kadrovsku evidenciju, materijalno poslovanje škola, kao i za ostale potrebe.

Preovlađuje izgleda stav da je dovoljno nabaviti računar, a da gotovi programi dolaze sami od sebe i da ne koštaju ništa. Razvijeno piratstvo u oblasti igara za računare, koje popularišu na žalost i računarski časopisi, kao da su stvorili utisak da je softver besplatan. Iskustva iz inostranstva govore da se u softver ulaže desetostruko više nego u hardver.

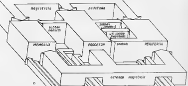
U periodu od 1970. do 1986. godine naučno-istraživačkim radom o primeni računara u obrazovanju bavili su se u Beogradu Prirodno-matematički fakultet, Multidisciplinarni centar Univerziteta, Elektrotehnički fakultet, Računarska laboratorija Instituta „Boris Kidrič", Institut „Mihajlo Pupin" i Računarska laboratorija u Republičkom zavodu za unapređivanje vaspitanja i obrazovanja.

1 JEDAN POGLED IZNUTRA

Organizacija računara

Jovan Skuljan

Kućni računari su prvo osvojili naša domova, a sada, evo, osvajaju i naša škola. Sve je viša korisnika računara i sva viša potrebe da se sazna „šta to ima u kutiji" prad kojom sedimo ponekad i satima. U želji da pomognamo i nastavnicima i učenicima, posvetićemo pažnju pitanju organizacija kućnog računara, objedinjujući na neki način sve ono o čemu smo u „Računarima" već i pisali.



Slika 1 Blok šema računara

Računar je složen sistem ali principi na kojima radi veoma su jednostavni Istina tržište kućnim računarima danas je preplavljeno najrazličitijim modelima koji ne samo da izgledaju različito (ko bi rekao da spektrum i XT rade isti posao?), već praktično ne postoje ni dva računara koji mogu koristiti iste programe Međutim to su sve spoljašnji efekti Duboko unutra na nivou elektronskih kola računar je bio (i ostaće bar još neko vreme) sistem baziran na nekim univerzalnim (rekli bismo „klasičnim") principima

Blok šema tipičnog računara data je na slici 1

Srce računara čini *mikroprocesor* — sklop koji izvršava instrukcije zadate *programom* i povezuje sve ostale komponente sisteme

Memorija ima ulogu privremenog skladišta informacija dok *periferni uređaji* predstavljaju vezu računara sa svetom

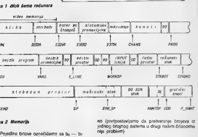
Veza između mikroprocesora memorije i periferije ostvaruje se preko posebnih električnih linija zvanih *magistrale* Razlikujemo tri magistrale prema njihovoj nameni i to su *adresna magistrala*, *magistrala podataka* i *kontrolna magistrala*

Memorija

Memoriju je najlakše shvatiti kao dugačak niz pregradaka (ćelija) u koje procesor može da upisuje sadržaje i iz kojih može te sadržaje da čita (sl 2)

Mi ćemo za memorijske ćelije koristiti termin *bajt* Svaki bajt ima svoju *adresu* — ceo broj koji jednoznačno određuje njegov „položaj" u memoriji Kada mikroprocesor želi da komunicira sa nekim bajtom mora da postavi odgovarajuću adresu na adresnu magistralu i tek tada da prozove memoriju

Jedan bajt se sastoji od osam *bita* — elementarnih memorijskih ćelija, koje su u stanju da upamte samo jednu od dve osnovne informacije Bit je *resetovan* ako mu je sadržaj nula ili je *setovan* ako mu je sadržaj jedinica (sl 3)



Slika 2 Memorija

Pojedine bitove označićemo sa b_0 — b_7

Stanje nekog bita fizički se karakteriše *naponom* na izlazu odgovarajuće ćelije Čitav računar se bazira na tzv *logičkim kolima* koja rade sa isključivo dva nivoa električnog napona 0V označava *logičku nulu* a +5V *logičku jedinicu*

Postoji tačno $2^8=256$ različitih načina da se nule i jedinice raspodele na osam slobodnih mesta (u matematici bismo to definisali kao *varijacije sa ponavljanjem osme klase od dva elementa)* To znači da jedan bajt može nositi 256 različitih informacija

Ako sadržaj bajta (niz od osam nula i jedinica) shvatimo kao broj u *binarnom sistemu* onda možemo reći da jedan bajt sadrži vrednost između *00000000* i *11111111* što odgovara rasponu 0—255 dekadno ili 00—FF heksadekadno (pretpostavljamo da pretvaranje brojeva iz jednog brojnog sistema u drugi našim čitaocima nije problem)

ROM i RAM

U zavisnosti od fizičke realizacije strukture memorije može biti takva da mikroprocesor jedino ima pristup čitanju sadržaja, ili takva da mikroprocesor može po volji da upisuje i čita sadržaje Memorije iz koje se može samo čitati naziva se ROM *(Read Only Memory)*. Zapravo procesor sasvim slobodno može pokušati da upiše nešto ali će ta akcija ostati bez efekta Sadržaj ROM-a ostaje neizmenjen čak i ako isključimo računar Deo memorije računara je uvek organizovan kao ROM i tu su smešteni neki fundamentalni programi i podaci bez kojih sistem naprosto ne bi mogao da funkcioniše

Drugi tip memorije sa proizvoljnim pristu-

pom nazivamo RAM *(Random Access Memory)*. Tako je organizovan najveći deo memorije računara i sloj na raspolaganju korisniku. Prestankom napajanja čitav sadržaj RAM-a se nepovratno gubi.

Periferija

Periferiju praktično čine svi uređaji priključeni na računarski sistem spolja: tastatura, štampač, monitor, disk, miš, kasetofon itd. Svi ti uređaji prvenstveno omogućuju računaru da stupi u kontakt sa spoljašnjim svetom, odnosno sa čovekom kao korisnikom.

Preko tastature ili miša korisnik unosi svoje zahteve računaru. Preko ekrana i štampača računar prezentira rezultate svog rada u formi koje korisnik prepoznaje i može da upotrebi. Konačno, kasetofon (na manjim računarima) i disk (na onim malo većim) služe za trajno čuvanje informacija.



Slika 3 Bajt

Adresna magistrala

Kada mikroprocesor želi da stupi u vezu sa nekom komponentom računarskog sistema, on mora na neki način da obavesti čitav sistem o tome koja ga komponenta konkretno interesuje (koji bajt memorije ili koji periferijski uređaj). Slikovito rečeno, mikroprocesor mora da *adresira* željenu komponentu sistema. A to postiže tako što odgovarajuće električne nivoe dovodi na linije tzv. *adresne magistrale*.

U jednom trenutku jedna linija magistrale može biti ili na naponu 0V (logička nula) ili na naponu +5V (logička jedinica). Svaka kombinacija nula i jedinica na magistrali predstavlja adresu jednog bajta u memoriji ili jednog uređaja na periferiji. Zapravo, kada se radi o periferiji moramo biti obazrivi, jer svaki periferijski uređaj je previše složen da bi mikroprocesor s njim mogao komunicirati preko *samo jedne* adrese. Uostalom, sa gledišta mikroprocesora periferiju i ne čine nikakvi mehanički uređaji. Ne vidi procesor štampač, na primer, kao mehaničku bučnu kutiju koja lupa slova po papiru. Umesto toga, postoji par adresa na koje procesor može nešto da upisuje i sa kojih može da čita — i to je sve. Periferija procesoru izgleda kao memorija. I tamo postoje nekakvi bajtovi i svaki od tih bajtova ima svoju adresu. Svaki takav bajt označavamo kao *I/O port (INPUT/OUTPUT Port)*. Da bi adresirao neki port, procesor mora da dovede odgovarajuću kombinaciju nula i jedinica na adresnu magistralu i da onda prozove periferiju.

Broj linija na adresnoj magistrali zavisi od tipa procesora. Na primer, Z80 ima 16 linija, a 8088 čak 20 linija.

Magistrala podataka

Razmena informacija između mikroprocesora i ostalih elemenata računarskog sistema memorije i periferije vrši se preko posebnih električnih linija koje čine tzv. *magistralu podataka*.

Kod većine kućnih računara magistrala podataka se sastoji od osam linija, pri čemu svaka linija kao i do sada može biti na naponu 0V ili +5V. Procesor može da postavi napone na magistralu podataka proizvoljno, i tada kažemo da *on upisuje* sadržaj. Taj sadržaj pročitaće neki od periferijskih uređaja ili će biti upisan u neku od memorijskih lokacija.

Isto tako, procesor može da *pročita* sadržaj magistrale, u kom slučaju upisivanje vrši neka od preostalih komponenti sistema. Zato kažemo da je magistrala podataka *dvosmerna*.

Što se tiče adresne magistrale, ona je jasno *jednosmerna*, jer jedino procesor vrši adresiranje.

Kontrolna magistrala

Kontrolnu magistralu čine linije sa specijalnom namenom, preko kojih procesor stavlja na znanje svim ostalim komponentama u sistemu šta upravo radi ili namerava da uradi. Isto tako, preko kontrolnih linija mikroprocesor prima informacije o tome šta ostali uređaji rade.

Linije *„zahtev memoriji“* i *„zahtev periferiji“* koristi procesor kada želi da prozove neku memorijsku lokaciju ili neki periferijski uređaj. Dok su ove linije neaktivne, napon na njima je +5V. Kada proziva memoriju, procesor dovodi 0V na liniju *„zahtev memoriji“*, a kada proziva periferiju, dovodi 0V na liniju *„zahtev periferiji“*.

Slično tome, linijama *„upis“* i *„čitanje“* mikroprocesor naznačuje koju od dve operacije želi da izvede. Na primer, ako želi da upiše nešto u memoriju, procesor će dovesti 0V na linije *„zahtev memoriji“* i *„upis“*.

Neki mikroprocesori, kao što je 8088, imaju po jednu kontrolnu liniju za svaku od pomenutih akcija: *„upis u memoriju“*, *„čitanje iz memorije“*, *„upis na periferiju“* i *„čitanje sa periferije“*, ali to nije nikakva suštinska razlika. Uostalom, kontrolne linije Z80 mogu se primenom par jednostavnih logičkih kola transformisati tako da budu identična sa onima koja ima 8088.

Memorijska mapa

Skup svih memorijskih lokacija kojima mikroprocesor može da pristupa čini tzv. *memorijski adresni prostor* ili *memorijsku mapu*. Adresni prostor određen je tipom procesora, odnosno brojem električnih linija na adresnoj magistrali.

Pošto svakoj memorijskoj lokaciji odgovara jedna adresa, adresni prostor je jednak ukupnom broju svih različitih rasporeda nula i jedinica na *n* linija magistrale. Ako se ponovo pozovemo na srednjoškolsko znanje matematike, zaključićemo da se radi o *varijacijama sa ponavljanjem n-te klase* od *dva elementa*, a njih ima tačno 2^n.

Procesor Z80, na primer, ima 16 linija na adresnoj magistrali i prema tome može da adresira $2^{16}=65536$ bajtova, odnosno 64 kilobajta (kilobajt K ima 1024 bajtova, ili 2^{10}).

Slično tome, procesor HD64180, o kome smo pisali u „Računarima 37“, ima adresni prostor od 512K, jer se obraća memoriji preko 19 adresnih linija ($2^{19}=2^9\times2^{10}=512\times2^{10}$).

Konačno, procesor 8088 ima 20-bitnu adresnu magistralu, što mu omogućuje da neposredno adresira 1024K (jedan megabajt) memorijskog prostora.

I/O mapa

Pri prozivanju periferije, mikroprocesor postupa praktično na isti način kao i kada pristupa memoriji. Adresnom magistralom se prenosi adresa uređaja, a magistralom podataka teku informacije.

Skup svih spoljašnjih uređaja kojima procesor može da pristupa čini tzv. *I/O adresni prostor* ili *I/O mapu*. Veličina tog prostora opet je određena brojem adresnih linija koje mikroprocesor koristi u komunikaciji sa periferijom.

Pod pojmom „periferijski uređaj“, naravno, ovde podrazumevamo uređaj *onako kako ga doživljava sam mikroprocesor*, a to praktično znači *I/O port*. Jedan složeni uređaj, već smo to rekli, može imati više I/O portova i prema tome zauzimati više I/O adresa.

Mikroprocesor Z80 koristi svih 16 adresnih linija za prozivanje periferije, pa tako ima I/O adresni prostor od 64K. Međutim, I/O mapa je potpuno nezavisna od memorijske mape, jer se za njihovo prozivanje koriste različite kontrolne linije, o čemu ćemo još govoriti.

Intelov 8088, s druge strane, od 20 adresnih linija koristi samo 16 za komunikaciju sa periferijom. Adresni prostor je, dakle, takođe 64K.

Neki mikroprocesori, kao što je 6502, *uopšte nemaju* zasebnu I/O mapu, već sve periferijske uređaje (I/O portove) tretiraju bukvalno kao memorijske lokacije. Uređaji su priključeni na memorijsku mapu i sa njima procesor komunicira kada god prozove odgovarajući bajt u memoriji.

Prozivanje memorije

Kada kažemo „procesor komunicira sa memorijom“, tu mnogo stvari ostaje nejasno. Kako pre svega mikroprocesor uspostavlja vezu sa određenim bajtom u memorijskoj mapi?

Kao što smo već rekli, sve se odvija na bazi elektronike logičkih kola. Signali koji putuju magistralama imaju fiksne logičke nivoe, a sadržaj magistrale može se shvatiti kao binarni broj.

Recimo da treba prozvati memorijsku lokaciju 37251 i u nju upisati sadržaj 197. Pre svega, procesor mora da postavi adresu na adresnu magistralu. Ako magistrala ima 16 linija, onda će za zadatu adresu njen sadržaj biti *1001 0001 1000 0011*. Na neke linije magistrale procesor će postaviti napon 0V, a na druge napon +5V.

Zatim mikroprocesor dovodi napon od 0V na linije *„zahtev memoriji“* i *„upis“*, a na magistralu podataka dovodi broj 197, odnosno kombinaciju *1100 0101*.

Sama memorija reaguje na sledeći način.

Električni sadržaj adresne magistrale dopušta samo jednoj memorijskoj ćeliji (onoj čija je adresa zadata) da stupi u vezu sa magistralom podataka. Signal *„zahtev memoriji“* aktivira memoriju i povezuje prozvanu ćeliju sa magistralom podataka. Signal *„upis“* stavlja memoriju u stanje iščekivanja, odnosno prijema podataka. Sadržaj magistrale podataka koji pristigne u tom trenutku upisuje se u prozvani bajt i proces je završen. Procesor vraća liniju *„zahtev memoriji“* na nivo logičke jedinice (neaktivno) i memorija mirno čeka eventualnu sledeću prozivku, dok procesor radi svoj posao.

Kasnije će, recimo, procesor poželeti da pročita sadržaj bajta 37251. Postaviće ponovo istu adresu na magistralu, dovešće 0V na linije *„zahtev memoriji“* i *„čitanje“* i čekaće da magistralom podataka stigne odgovor. Sama memorija dopušta prozvanoj ćeliji da svoj sadržaj postavi na magistralu podataka. Procesor prihvata taj sadržaj, unosi ga u neki od svojih registara, dovodi +5V na liniju *„zahtev memoriji“* i nastavlja sa radom.

Prozivanje periferije

Pristup periferiji identičan je prozivanju memorije s tom razlikom što se umesto linije „zahtev memoriji" koristi linija „zahtev periferiji". To znači da kada procesor komunicira sa periferijskim uređajem na nekoj adresi 753 bajt na adresi 753 sa tim *nema nikakve veze* jer memorija pri tome ostaje neaktivna (govorimo o procesorima sa razdvojenom memorijskom i I/O mapom).

Uzećemo, na primer, komunikaciju sa štampačem na računaru „PC XT". Mikroprocesor 8088 nije direktno vezan za štampač već preko posebnog kontrolera, koji ima tri I/O porta. To znači da mikroprocesor doživljava štampač kao *tri periferijska uređaja* na tri različite I/O adrese 3BC, 3BD i 3BE.

Registar na adresi 3BC prenosi podatke štampaču (ASCII znakove koji se štampaju i komande za upravljanje procesom štampanja). Da bi poslao štampaču neko slovo, recimo „A", mikroprocesor mora:

1. Da postavi adresu 3BC na adresnu magistralu.
2. Da postavi slovo „A" (ASCII kod 65 decimalno) na magistralu podataka.
3. Da aktivira liniju „*upis na periferiju*".

Periferija će reagovati tako što će registar na I/O adresi 3BC prihvatiti slovo „A" i proslediti ga štampaču.

Slično tome, registar na I/O adresi 3BD nosi informaciju o stanju štampača (tzv. *statusni registar*). Očitavanjem tog registra mikroprocesor može da sazna da li je štampač spreman da primi novi znak, da nije možda nestalo papira itd.

Ustupanje magistrala

U složenijim računarskim sistemima može se javiti potreba da jedan periferijski uređaj poželi *direktno da pristupi memoriji* bez posredovanja mikroprocesora ili, možda, jedan periferijski uređaj treba da komunicira sa drugim, opet bez mešanja procesora u sve to.

Ovakvi zahtevi obično su vezani za uštedu u vremenu i mora im se posvetiti pažnja.

Jasno, ako neki uređaj želi da komunicira sa nekom od komponenti sistema, on mora vršiti *adresiranje*, tj. upisivati nešto na adresnu magistralu. Da bi to uopšte bilo moguće, mikroprocesor o tome *mora biti obavešten* jer, već smo to rekli, on drži monopol nad adresnom magistralom, i uvek on nešto tu upisuje. Da ne bi došlo do konfuzije, procesor mora da se „isključi" sa adresne magistrale i da je prepusti drugima.

Ustupanjem samo adresne magistrale malo bi se šta postiglo. Svaka komunikacija podrazumeva i korišćenje magistrala podataka za razmenu informacija, kao i korišćenje kontrolnih linija. Prema tome, procesor treba da ima mogućnost potpunog ustupanja magistrala periferijskim uređajima.

Proces ustupanja magistrala protiče na sledeći način:

Uređaj koji to želi prvo mora uputiti odgovarajući zahtev procesoru, aktiviranjem linije „*zahtev za magistralom*". Ta linija je, inače, sve vreme na nivou logičke jedinice (neaktivno). Njeno aktiviranje vrši se dovođenjem na 0V.

Kada primi zahtev, procesor privede kraju posao koji trenutno obavlja, a onda se isključi sa svih magistrala i pošalje napolje signal „*magistrale slobodne*". Uređaj koji je magistrale tražio to spremno dočeka i počne da adresira po svojoj volji.

Program

Sve što smo do sada rekli o računarskom sistemu uglavnom bi se podvelo pod odrednicu „kako sve to izgleda". Isto je tako, međutim interesantno i pitanje „kako sve to radi?"

U redu, znamo da procesor proziva memoriju tako što aktivira adresnu magistralu i liniju „*zahtev memoriji*". Ali kako on zna *kada šta treba da uradi?* Zašto on uopšte u nekom trenutku proziva neku memorijsku lokaciju? Šta zapravo definiše rad jednog računara?

Odgovor na to pitanje glasi kratko: *program*.

Program je zapisan u memoriji i predstavlja *niz operacija* koje mikroprocesor izvršava. Mikroprocesor čita redom bajt po bajt memorije i shvata to kao nekakve *naredbe*, koje ima da izvrši.

Međutim, da pođemo od početka.

Osim strujnog napajanja, koje je neophodno za električno funkcionisanje mikroprocesora, postoji još jedna ulazna linija za tzv. *takt* ili *klok (clock)*. Klok je pravilan periodični signal sa naizmeničnom promenom nivoa od nule do jedinice i obratno (promena napona između 0V i +5V). Elektronski sklop mikroprocesora reaguje na svaku promenu nivoa kloka i to, zapravo, predstavlja stvarnu *pogonsku snagu*, kao što voda pokreće vodenični točak, ili kao što vetar pokreće krila vetrenjače.

Kada ne bi bilo kloka, procesor ne bi radio apsolutno ništa. Što je brži klok, procesor radi sve brže, ali, nažalost, to ne može ići unedogled. Tipična vrednost frekvencije kloka za popularne mikroprocesore je reda *nekoliko megaherca* (nekoliko miliona promena nivoa u sekundi).

Čim se uključi napajanje i procesor se *resetuje* (poništi), i čim počne da radi klok, procesor automatski radi sledeće: dovodi nulu na čitavu adresnu magistralu (govorimo konkretno za Z80, ali slično je i kod drugih mikroprocesora) i zatim aktivira signale „*zahtev memoriji*" i „*čitanje*". Sve to skupa znači da procesor želi da pročita sadržaj *prvog bajta memorije* (adresa 0). Na tim niskim adresama je obično ROM, sa tzv. *sistemskim programom*. Prvi bajt memorije je ujedno i prva naredba koju će procesor izvršiti.

Kada privede kraju izvršenja prve naredbe, mikroprocesor automatski (gonjen klokom) čita sadržaj sledećeg bajta memorije i shvata to kao novu naredbu koju takođe izvršava. Proces tako teče sve dok se računar ne isključi.

Same naredbe programa su, po pravilu, veoma jednostavne. Najčešće je to neko premeštanje sadržaja iz jedne memorijske lokacije u drugu, slanje raznoraznih sadržaja na periferiju, sabiranje dva sadržaja nekih registara i slično. Međutim, kada se tako jednostavne naredbe slože u veliki program od nekoliko hiljada bajtova, računar počinje da obavlja vrlo složene poslove, u šta smo već svi imali prilika da se uverimo.

Prekidi

Komunikacija procesora sa periferijom dosta je složena iz više razloga. Mada smo do sada, uglavnom isticali analogiju između memorije i periferije, radi se o ipak suštinski različitim komponentama sistema.

Memorija je *pasivna* i procesor joj pristupa kada i kako to želi. Memorija je dalje *veoma brza* i procesor ne gubi vreme dok s njom komunicira.

Periferijski uređaji s druge strane mogu biti i *aktivni*. Oni često imaju šta da kažu procesoru a imaju čak i šta da zahtevaju od procesora. Uglavnom su *veoma spori* i samo rasipaju procesorsko vreme.

Analogija između periferije i memorije je, dakle, samo u načinu pristupa i adresiranja od strane mikroprocesora.

Ako procesor želi da prenese u memoriju veći blok bajtova, on će to uraditi nizom naredbi za upis, izvršavajući ih jednu za drugom. Memorija je dovoljno brza da odmah prihvati to što procesor upisuje.

Ali ako mikroprocesor hoće da pošalje štampaču nekoliko slova za ispis, biće problema. Dok štampač primi jedno slovo, pa dok pokrene



Slika 4. Prioriteti u sistemu prekida

mehanizam za štampanje, pa dok bude spreman za sledeće slovo, prođe gotovo čitava večnost ako gledamo sa strane jednog mikroprocesora.

Istina, ni jedan štampač ne prima od mikroprocesora samo jedno slovo. Prima ih više u blokovima. Ali i dalje procesor ima da čeka dok štampač ne ispiše sva slova iz primljenog bloka. A svako čekanje je gubljenje vremena.

Vreme se gubi, recimo, i u radu sa tastaturom. Manji računari, kao što je „spektrum", rade tako što mikroprocesor povremeno (pedeset puta u sekundi) *prekida svaki posao* koji trenutno radi (tekući program) da bi očitao tastaturu i video da li je tamo neko nešto pritisnuo. Zatim se procesor vraća na prekinuti program i nastavlja posao tamo gde je stao.

Izgleda prilično nerazumno. Zamislimo samo kada bismo mi po ceo dan povremeno dizali telefonsku slušalicu, tek da vidimo da li nas neko traži (?!) U tom slučaju morali bismo vrlo često da prekidamo svoj posao, što je i gubljenje vremena i gubljenje živaca (mikroprocesor ima jedino tu prednost što ne gubi živce).

Naravno, problem sa telefonom je rešen tako što postoji zvučni signal koji označava da nas neko traži i tada moremo, *ali sa razlogom*, da prekinemo posao.

Zar ne bi mogla tastatura računara da bude makar malo inteligentnija, pa da sama javi mikroprocesoru kada je neki taster pritisnut? U međuvremenu mikroprocesor može mirno da izvršava tekući program. Ili, zar ne bi štampač mogao da javi procesoru kada završi ispis bloka znakova, a da procesor dotle radi po programu bez prekida i čekanja?

Odgovor je, naravno, da *može* i tako rade svi mikroprocesori. Jedino se te mogućnosti u jeftinijim kućnim računarima ne koriste u potpunosti.

Postoje posebne električne linije mikroprocesora na koje pristižu zahtevi za *prekid (interrupt)*. Čim se na liniji za prekid pojavi logička nula, to je znak da neki periferijski uređaj nešto traži. Procesor samo privede kraju naredbu koju upravo izvršava, i odmah gleda ko ga je prekinuo i zašto.

Kad primi zahtev za prekid, procesor u stvari prelazi na izvršenje *posebnog programa za obradu prekida* (servisna rutina). Svaki uređaj ima svoju servisnu rutinu i do zabune ne može doći. Može samo doći do gužve ako više uređaja istovremeno traži prekid, ili jedan uređaj prekida izvršenje servisne rutine drugog uređaja. Da bi se takve stvari regulisale, uvodi se pojam *prioriteta prekida*. Naime, tačno se zna ko koga može da prekida i čiji su zahtevi važniji.

Prioriteti se mogu ostvariti posebnim vezivanjem uređaja u jedinstveni lanac, kao što prikazuje slika 4.

Uređaj sa najvišim prioritetom (A) priključuje se direktno na procesorsku liniju „*zahtev za prekid*". Uređaj B, koji ima niži prioritet, vezuje se na uređaj A. To znači da kada B traži prekid, taj zahtev će stići do procesora samo ukoliko A to dopusti, tj. ukoliko A ne traži prekid. Takođe sledi da uređaj A može da prekine servisnu rutinu uređaja B, dok uređaj B ne može da prekida A.

Lanac uređaja može biti proizvoljno dugačak. Poslednji uređaj u lancu ima najniži prioritet. Njegov zahtev za prekid će stići do procesora samo ako ni jedan od ostalih uređaja sa višim prioritetom istovremeno ne traži prekid.

Dejan Ristanović

2 Periferijska oprema

JEDINICE OD KOJIH NE BOLI GLAVA

Sam po sebi, svaki je kompjuter gluvonem i slep — mikroprocesor može da obrađuje podatke, ali mu te podatke treba nekako dostaviti. Slično tome, da bi obrada podataka imala ikakvog smisla, računar dobijene rezultate treba da saopšti čoveku. Periferijska oprema preuzima na sebe ova dva osnovna i mnoge druge zadatke. Mogło bi se, dakle, reći da periferijska oprema predstavlja vrata kroz koja računar komunicira sa spoljnim svetom.

U periferijsku opremu ubrajamo razne ulazne (tastature miš džojstik skener svetlosno pero optička tabla uređaj za prepoznavanje glasa i slično) izlazne (monitor štampač ploter sintetizator govora itd) i mešovite (modem) jedinice jedinice spoljne memorije (flopi hard i laserski diskovi CD ROM-ovi i tome slično) i mnoge druge dodatke koji se na tržištu svakodnevno pojavljuju i kojih ima dovoljno da popune mnogo specijalnih izdanja ovog formata Zato ćemo u okviru ovoga teksta upoznati samo osnovne periferijske jedinice koje su u praktičnom radu najpotrebnije

Ulazne jedinice

Iako su skeneri, svetlosna pera, grafička tabla i slični uređaji iz dana u dan sve popularniji, veći deo podataka se i dalje unosi posredstvom stare, dobre, tastature koja, dakle, predstavlja osnovnu ulaznu jedinicu svakog personalnog računara.

Tastatura i miš

Tastatura personalnog računara unekoliko podseća na električnu mašinu — dirke sa slovima su raspoređene na isti način, pridodata je numerička tastatura, prepoznaju se i mnoge značajne kontrolne dirke Računarska tastatura su, ipak mnogo raznovrsnije — na tržištu je čitav spektar tastatura raznih tipova koje se, mada na prvi pogled iste razlikuju po kvalitetu trajnosti i ceni

Dobre tastature su *mehaničke* premda ni one nisu sasvim iste IBM-ovi računari su poznati po izvanrednim mehaničkim tastaturama dok neke kopije IBM-a mogu da imaju tastature koje su na prvi pogled sasvim slične ali daleko slabijeg kvaliteta (zato su jeftinije) Za obradu teksta je važna i takozvana *numerička* tastatura koja se koristi kod dugotrajnog kucanja brojeva pogodnije je da cifre budu u tri reda tako da su pristupačne prstima jedne ruke Tu su najzad i *funkcijski* tasteri kojima autori raznih komercijalnih programa dodeljuju specijalne funkcije kao što je umetanje slova premeštanje paragrafa pretraživanje baze podataka i slično

Membranske tastature se na prvi pogled ne razlikuju od mehaničkih tasteri su slični ali se kontakt ostvaruje pritiskom na membranu koja je skrivena ispod dirki U teoriji su membranske tastature lošije od mehaničkih ali se to ponekad slabo primećuje — razlika je obično u veku trajanja Neki profesionalni sistemi za obradu teksta najzad imaju *kapacitivnu* tastaturu koja je obično izuzetno kvalitetna

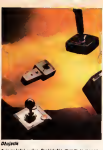


Optički miš

Standardna oprema mnogih personalnih računara obuhvata i takozvanog *miša* uređaj koji pomerate po stolu dajući računaru signale o operacijama koje treba da izvrši Miševi su izmišljeni zbog početnika kojima je bilo teško da pamte sintaksu pojedinih komandi — mnogo je lakše odvesti kurzor do sličice koja označava željenu operaciju i pritisnuti neki taster Miševe danas upotrebljavaju i iskusni korisnici računara ubrzavajući na taj način neke operacije Obzirom da je ovaj relativno jeftini dodatak neophodan raznim komercijalnim programima za crtanje i projektovanje svakako ga treba nabaviti

Džojstik

Palica za igru ili džojstik je sasvim prirodno popularna kod ljubitelja arkadnih igara — vrhunski rezultati se teško mogu postići udaranjem po tastaturi! Razlikujemo dva tipa palica za igru analogni džojstik je skup i zahteva specijalni AD konvertor uz pomoć koga računar precizno odre-



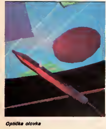
Džojstik

đuje položaj palice. Prekidački džojstik je mnogo jednostavniji — četiri prekidača i opruga koja vraća palicu u centralni položaj. Analogni džojstici su pogodniji za razne simulacije vožnje ili letenja, dok su prekidački idealni za prave „pucačke" igre. Većina džojstika je kompatibilna sa standardima koje je propisao Atari, što podrazumeva i precizno definisan konektor — isti Atari kompatibilan džojstik možete da priključite na „atari ST", „amigu", „komodor 64", „amstrad spektrum" opremljen odgovarajućim interfejsom.

Od linije do memorije

Brojeve je lako otkucati, ali je unošenje grafičkih oblika u memoriju računara priličan problem. Pod grafičkim oblicima ovde ne podrazumevamo grafike funkcija — ništa lakše nego programirati računar da grafik zadate funkcije prikaže na ekranu. Ukoliko je, međutim, crtež urađen slobodnom rukom ili ako je toliko složen da se ne može razložiti na formalno definisane segmente, na scenu stupaju grafičke table i optičke olovke.

Izgled grafičke table iznenadiće malo koga — plastična podloga na koju se može staviti špartani papir i olovka (zvana *stylus*) kojom se po tom papiru piše. Stvar ipak nije baš tako jednostavan — olovka je povezana sa računarom koji



Optička olovka

može veoma precizno (preciznost jasno zavisi od cene) odrediti njen položaj i prema tome koordinate upravo stavljene tačke. Pri tome se koriste već pomenuti AD konvertori — analogni položaj pisaljke pretvara se u koordinate koje moraju da budu digitalne da bi ih kompjuter „razumeo". U zavisnosti od principa na kome se zasnivaju, razlikujemo kapacitivne (preciznost 0,02 mm) i magnetske grafičke table: cena ovih uređaja retko silazi ispod 100 funti, dok su profesionalne grafičke table ponekad skuplje od solidnih personalnih računara.

Svetlosna pera su mnogo jeftinija — uz njihovu pomoć povlačimo linije po samom ekranu našeg televizora ili monitora, a računar preuzima podatke i te linije trenutak docnije zaista iscrtava. Princip delovanja optičkog pera je ingeniozno smišljen a ipak krajnje jednostavan: obzirom da sliku na ekranu iscrtava precizno usmereni snop elektrona, fotoćelija na vrhu pera otkriva nailazak snopa a računar, koristeći interni časovnik, jednostavno proračunava koordinate. Jasno je da ovaj metod nije mnogo precizan i da je zbog toga osetljivost a samim tim i primenjivost svetlosnih pera manja od osetljivosti i primenjivosti grafičkih tabli. U poslednje vreme pojavili su se i ekrani osetljivi na dodir (umesto da kucamo broj opcije koja nam je potrebna, prstom dodirujemo njen opis) čija je preciznost, jasno, dalje smanjena.

Skaneri

Grafička tabla i optička pera u poređenju sa skanerima predstavljaju dečje igračke — skaner je uređaj koji precizno digitalizuje kompletnu sliku i prenosi je u memoriju računara na dalju obradu. Na taj način ilustracije iz jednog štampanog teksta možemo preneti u tekst koji upravo pripremamo, uneti potrebne izmene i rezultate štampati na laserskom printeru. Ukoliko izostavi-



Digitalni skener

mo obradu, baterija skaner — laserski štampač predstavlja mašinu za foto-kopiranje.

Prilikom skaniranja slike glavni parametar je rezolucija: modeli umerenih cena obezbeđuju više nego solidnih 600 tačaka po inču, pri čemu se rezolucija može po volji smanjivati u cilju uštede memorije. Skaneri su inače veliki gutači memorije i procesorskog vremena — standardan IBM PC XT opremljen hard diskom od 20 megabajta može da „zapamti" svega desetak A4 slika, pri čemu digitalizacija svake od njih traje tridesetak sekundi.

Pošto je slika u memoriji ili na disku, apetiti rastu: zar ne bi bilo zgodno da računar „pročita" tekst i pripremi ga za obradu? Problem prepoznavanja oblika (pa samim tim i oblika slova) je do skora smatran tipičnim primerom takozvane veštačke inteligencije — danas se na tržištu nalaze brojni optički čitači koji mogu da transformišu štampani tekst u ASCII datoteku i programi koji „čitaju" tekst sa digitalizovane slike. Većina jeftinih optičkih čitača, na žalost, i dalje predstavlja samo igračke.

Modemi

Modem ne predstavlja samo ulazni uređaj — radi se o napravi uz pomoć koje dva geografski udaljena računara mogu da razmenjuju podatke posredstvom standardnih telefonskih linija. Na Zapadu su veoma popularne kompjuterske mreže — pretplatnik se posredstvom modema i telefona povezuje sa centralnim računarom, koristi njegove usluge, ostavlja poruke drugim korisnicima, obavlja razne administrativne, bankarske i poštanske poslove i tome slično. U Jugoslaviji su modemi relativno retki jer njihova upotreba nije zakonski definisana — pošta jednostavno ignoriše zahteve da se ovakvi uređaji atestiraju!

Glavni parametar modema je brzina prenosa — obično se podaci primaju i šalju brzinom od 300 bauda (300 bita u sekundi), što znači da prenos jedne šlajfne teksta traje nešto manje od minuta. Pri komunikaciji sa mrežom smatra se logičnim da korisnik šalje malo pitanja i dobija mnogo odgovora, pa se podaci emituju brzinom od 300 a primaju brzinom od 1200 bauda. To je ujedno i opravdanje što smo modeme pomenuli u okviru „ulaznih jedinica".

Spoljna memorija

Podaci upisani u memoriju personalnog računara obično se nepovratno uništavaju posle svakog prekida u napajanju. Ukoliko, dakle, želimo da sačuvamo programe i podatke dok je računar isključen, moramo ih prepisati na neki medij spoljne memorije. Pri tom se najčešće koriste magnetni mediji: kasete, diskete ili hard disk.

Kasetofon

Vlasnici prvih kućnih računara su programe, podatke i tekstove upisivali na kasete, koristeći pri tom bilo koji kasetofon koji se nađe u kući (dva časna izuzetka: u cenu „amstrada 464" je uračunat specijalni kasetofon koji, ako se opredelite za „komodor 64", morate posebno da dokupite). Dobra strana kasetofona su, dakle, što ga već imate i što su kasete relativno jeftin medij



Kasete

na svaku stranu C—60 kasete pri spektrumovoj brzini snimanja od 1500 bauda (1500 bits u sekundu) stane oko 320 K podataka. Loše strane su što je spor i nepouzdan: događaće vam se da ne možete da učitate programe koje ste sami snimili, dok su problemi sa učitavanjem programa i podataka koje su vam prijatelji snimili već poslovični. Rešenje je kupovina flopi disk jedinice.

Flopi disk jedinice

Umesto na kasete, flopi disk jedinice upisuju podatke na diskete, specijalne ploče od magnetnog materijala ugrađene u zaštitna kućišta od kartona ili plastike. Podaci se na disketu upisuju brzo, pronalaze se lako i stvar je sasvim kompatibilna: ako snimite program, moći će da ga učita svako ko poseduje kompjuter poput vašeg! Disk jedinica na kraju otvara puteve ka obradi podataka, odnosno rada sa *data base* programima.

Flopi disk jedinica može da bude jednostrana ili dvostrana — jednostrana ima samo jednu glavu, što znači da u jednom trenutku pristupa samo podacima koji su upisani sa jedne strane diskete. Dvostrana disk jedinica istovremeno može da pristupa podacima upisanim na obe strane diskete, što znači da na isti medij staje dva puta više informacija. Od vašeg računara i operativnog sistema zavisi da li ćete moći da formirate datoteku čiji se segmenti prostiru na obe strane diskete.

Svaka disketa je podeljena na koncentrične trake kojih obično ima 40 ili 80. Analogija sa gramofonskom pločom nije potpuna: trake na ploči predstavljaju neprekidan kanal kroz koji igla putuje od oboda do centra, dok su trake na disketi samo koncentrični krugovi — zamišljena igla bi se stalno vrtela po istom krugu. Svaka traka podeljena je, u zavisnosti od korišćenog standarda, na 8, 9, 10 ili čak 16 sektora, a svaki



Disk jedinice

sektor obezbeđuje upis 256 ili 512 bajta podataka.

Sve parametre jedne disk jedinice je najlakše shvatiti na primeru. Disk jedinica sa jednom glavom koja je pripremljena tako da može da podeli disketu na 40 traka može da primi oko 125 K neformatiranih ili oko 100 K formatiranih podataka. Izraz „formatirani podaci" zahteva dodatno objašnjenje: pre nego što počnemo da smeštamo podatke na jednu disketu, moramo da je formatiramo koristeći poseban program. U toku formatiranja računar proverava svaku traku na disketi, upisujući u svaku „ćeliju" neki kontrolni podatak i proveravajući korektnost upisa. Očigledna svrha ovoga postupka je proveravanje ispravnosti diskete na koju će svakako biti smešteni važni programi ili podaci. Početnici obično ne znaju da formatiranje ima i druge jednako značajne funkcije.

U toku formatiranja na početak svake trake bivaju upisani kontrolni podaci koji predstavljaju „čeksum" za tu traku. Šta beše čeksum? Kada upisujemo program na kasetu, na njegovom kraju biva upisan i jedan ili dva bajta pomoću kojih računar proverava ispravnost upisa i, ako nešto nije u redu, ispisuje *Tape loading error*, *Missed data*, *WHAT?* ili nešto slično. Sasvim ekvivalentnu funkciju ima i „čeksum" na disketi s tim što se, umesto prostog sabiranja bajtova, koristi daleko pouzdaniji CRC *(Cyclic Redundancy Check)*. Osim čeksuma uz svaku traku bivaju upisani i neki drugi podaci na kojima se nećemo zadržavati — za korisnika je dovoljno da zapamti da određen prostor ostaje neiskorišćen za njega. Ako se svaka traka sastoji od 10 sektora a svaki sektor obuhvata 256 formatiranih bajtova, na disketu staje $40 \cdot 10 \cdot 256/1024 = 100$ K podataka.

Pođimo korak dalje i omogućimo našoj disk jedinici da na svaku disketu upisuje 80 traka sa podacima, time smo, jasno, povećali kapacitet na 200 K. Mnogi početnici misle da 40 traka zauzima samo pola diskete i da jeftinije disk jedinice nisu u stanju da „dosegnu" drugu polovinu. Nije tako — kod skupljih disk jedinica glava je u stanju da se dvostruko preciznije pozicionira, pa je isti prostor bolje iskorišćen. Nastavljajući da proširujemo našu zamišljenu disk jedinicu, dodaćemo joj drugu glavu. Tako ćemo ponovo duplirati kapacitet — 400 K na svakoj disketi.

Mnogi smatraju da *double density* (dvostruko guste u bukvalnom prevodu) diskete imaju 160 umesto 80 traka. Ništa pogrešnije od toga — ovakvo pakovanje može da se primeni kako na disketama od 80 traka tako na disketama od 40 traka pa čak i na sada sasvim zastarelim disketama od 35 traka. Pri tome se količina informacija koja se smešta na disketu praktično duplira pri čemu vreme pristupa čak opada! Ukoliko, na primer, imamo disk jedinicu od 400 K i kupimo *double density* disk interfejs, radićemo sa oko 640 K formatiranog prostora; dobici se postižu povećanjem broja sektora na svakoj traci (npr. 16 umesto 10) i inteligentnijim pakovanjem podataka. Disk interfejsi ovoga tipa zasnovani su na savremenijim čipovima i obično umeju da „imitiraju" stare kontrolere, tj. da čitaju diskete snimljene uz pomoć *single density* interfejsa.

Karakteristiku disk jedinica koja se uočava na prvi pogled smo paradoksalno ostavili za kraj. Do pre desetak godina korišćene su isključivo disk jedinice od 8 inča („oklop" svake diskete je, dakle, bio kvadrat stranice od oko 20 cm), a onda su preovladale diskete od 5.25 inča. Poslednjih godina veliku popularnost stekle su takozvane mini-disk jedinice od 3.5 inča — mini-diskete su pakovane u čvrsta plastična kućišta što znači da su mnogo otpornije od standardnih. U ovom trenutku diskete od 3.5 inča ipak nisu idealno rešenje: većina programa se i dalje nabavlja na starim disketama a ne treba zaboraviti ni višu cenu svake diskete. Još je manje preporučljiva nabavka nestandardne disk jedinice od 3 ili čak 2 inča.

Hard disk

Sledeća je stepenica kupovina *hard diska*. Iako je po obliku sličan flopi disk jedinici, hard disk omogućava upisivanje 10—140 megabajta podataka, dok na jednu disketu obično staje manje od megabajta. Hard disk je, sa druge strane, obično nepomičan, dok diskete menjate — na hard disku se, dakle, drže važni programi i podaci koje trenutno obrađujemo, dok stvari koje smo završili premeštamo na diskete ili, ako se radi o stvarima koje vam *verovatno* nikada neće zatrebati, mnogo jeftinije trake. Prednost hard diska nije samo kapacitet — podaci upisani na njemu daleko se brže pronalaze i učitavaju, ali je i cena daleko viša.

Iako je količina podataka koje se smeštaju na



Hard disk

disketu u poslednjih nekoliko godina gotovo udesetostručena ugrađivanjem dodatne glave, povećanjem broja traka i uvođenjem dvostruke gustine upisa, flopi disk jedinice su 1985. i 1986. godine došle do svog vrhunca: pojavile su se fizičke prepreke koje praktično onemogućavaju povećanje njihovog kapaciteta. Kakve su to fizičke prepreke? Pre svega, flopi disk se okreće „svega" 300 puta u minutu, što znači da se podaci sa osamdeset traka moraju preneti u memoriju najmanje $80/300 = 0.26$ minuta ili oko 15 sekundi. U primenama u kojima se radi sa bazama podataka problemi se značajno uvećavaju: upisno-čitajuća glava treba intenzivno da se pomera između traka, što znači da će intervali u kojima čitav sistem čeka da se sektor koji treba pročitati nađe ispod glave postajati mnogo duži od vremena potrebnog za efektivno učitavanje podataka. Rezultat: neefikasnost.

Zašto disk ne bi mogao da se okreće brže? Možda vam je poznato da je upisno-čitajuća glava spuštena do površine diskete, što znači da između dve po prirodi različite površine postoji intenzivan fizički kontakt a samim tim i trenje koje se strahovito povećava ubrzavanjem diska. Veće trenje bi, jasno, dovelo do bržeg trošenja glave i same diskete: posle neke granične brzine bi magnetni sloj na disketi (a samim tim i podaci na njoj) bio jednostavno zbrisan!

Disketa je, kao što smo pisali, podeljena na četrdeset ili osamdeset traka, pri čemu ovaj broj ne može bitno da se poveća. Problem nije u tome što je teško preciznije pozicionirati glavu diska već u tome što se disketa ne nalazi uvek na istom mestu: ako dva puta umećemo disketu u drajv, nećemo je **jednako** snažno gurnuti niti ćemo **jednako** energično pritisnuti klapnu. Osim toga ne može se očekivati da će azimuti glava raznih drajvova biti **apsolutno** jednaki. Zato između traka mora da postoji dovoljan prostor da se glava može pozicionirati na svaku od njih uz potrebnu toleranciju. Ograničen broj traka povlači, jasno, i ograničen kapacitet diskete od (teorijski) najviše 2 megabajta.

Poslednje od ozbiljnih ograničenja flopi diskova je broj (kilo)bajtova koji mogu da se upišu na jednu traku: podaci se, kao što vam je verovatno poznato, upisuju promenom magnetnog fluksa kroz materijal. Diskete su, bez obzira na zaštićenost kartonskom kutijom i omotačem, izložene vazduhu koji je prepun čestica prašine, mora se nekako obezbediti da pad nekog takvog zrnceta na disketu ne bude poguban po podatke na njoj a takvo se obezbeđenje nikako ne slaže sa željom da se podaci na trakama pregusto pakuju!

Rešenje sva tri problema može da se dobije jedino suštinskom promenom koja se naziva hard disk. Ukoliko disketu zamenimo čvrstim magnetnim diskom koji se nikada neće vaditi iz drajva, problem preciznog pozicioniranja se rešava već u fabrici kada se uređaj fino šteluje. Ukoliko je disk zajedno sa upisno-čitajućom glavom hermetički zatvoren, nikakva spoljna prašina ne može da ugrozi podatke na njemu pa

je kapacitet jedne trake ograničen jedino karakteristikama materijala koji industrija može da proizvede

Ostao je još problem brzine okretanja diska njeno povećanje svakako skraćuje koristan vek uređaja. Trebalo bi dakle držati glavu na delić milimetra iznad površine diska (pri tom bi jasno trebalo pojačati sva magnetna polja) što bi moglo da se izvede pomoću veoma precizne mehanike. Kako međutim da držite glavu na delić milimetra iznad diska i da je pri tom brzo šetate od trake do trake? Za rešenje ovoga problema koje ne zahteva basnoslovne troškove je zaista trebalo dosta inspiracije ako se disk okreće dovoljno brzo vazduh koji se nalazi uz njegovu površinu će se kretati skoro jednako brzo. Ako je upisno-čitajuća glava dovoljno laka i aerodinamički oblikovana ona će plivati na tom sloju vazduha kao na nekom vazdušnom jastuku! Rastojanje između glava i površine diska danas varira između 2 mikrometra i 300 nanometara poređenja radi svaka dlaka na vašoj glavi je debela oko 70 mikrometara!

Kombinacija masivnog nepomičnog diska u hermetički zatvorenoj kutiji iznad koga pliva upisno-čitajuća glava je poznata pod imenom *Winchester disk*. Vinčester diskovi starijih generacija su mogli da se prepoznaju po priličnoj buci zvuk koji izaziva veoma brzo okrećući disk je unekoliko prigušen zvukom ventilatora koji se obavezno ugrađuje u kutiju. Ova je buka konstantna dok flopi disk počinje da se okreće tek kada je računar poželeo da mu pristupa hard disk se okreće neprekidno jer mu je potrebno nekoliko sekundi da postigne radnu brzinu od 3000 obrtaja u minuti i još nekoliko sekundi da se iz te pune brzine zaustavi

Zbog ovog ubrzavanja i usporavanja, procedura uključivanja računara koji su opremljeni hard diskovima ume da bude prilično komplikovana. Najpre se uključuje glavni prekidač a onda hard disk jedinica. Sledi podúže nestrpljivo posmatranje indikatora *Ready* posle čijeg paljenja smemo da uključimo i sam kompjuter. Neki proizvođači računara ugrađuju samo jedan prekidač na čitavu opremu a zatim uvode potrebna kola za kašnjenje tako da se računar automatski pali tek kada je disk operativan. IBM je taj problem rešio softverski modeli XT i AT su pravljeni tako da po uključenju dovoljno dugo proveravaju ispravnost i kapacitet svoje memorije

Pošto ste povezali hard disk sa računarom (potreban vam je i poseban interfejs jer hard diskovi ne mogu da koriste kontrolere za flopije) počnite da uživate u njegovim karakteristikama. Danas se standardni Vinčester diskovi od 5,25 inča okreću 3536 puta u minutu (uporedite ovo sa brzinom flopija od 300 obrtaja u minutu), pamte 10 20 40 80 ili 140 megabajte podataka (kod flopija ovaj broj retko prelazi 1,5 M) što zahvaljuju gustini pakovanja od 690 traka po inču (najviše 96 tpi kod flopije) i omogućavaju pristup svakom podatku za samo 8 ms (kod dobrih flopija vreme pristupa obično nije kraće od 100 ms). Brzina prenosa podataka je ponekad prevelika za neke osmobitne mikroprocesore pa se u hard diskova ugrađuju veliki baferi, što značajno povećava njihovu cenu. Alternativno rešenje je zamena programiranog ulaza-izlaza (koji se zasniva na mehanizmima takozvanog *hand-shaking*-a ili daleko češće, nemaskiranih prekida) mehanizmima direktnog memorijskog pristupa (DMA) kod koga mikroprocesor nema udela u prenošenju podataka između periferije i memorije

Vinčester diskovi su u celini gledano veoma pouzdani pa se često zanemaruje mogućnost gubitka podataka na njima. Čak i ako zanemarimo kvarove sasvim je moguće da ćete u nekom trenutku greškom nepovratno izgubiti rezultate svog višednevnog rada. Zato je veoma važno da s vremena na vreme prepišete čitav sadržaj hard diska na neki drugi medij obično na diskete za ovo prepisivanje će vam na žalost biti potrebno 10 20 40 ili možda 60 disketa (zavisno od kapaciteta hard diska i flopija koji posedujete)! Umetati dvadesetak disketa u drajv i njihovo snimanje je posao koji može da potraje više od sata što znači da se na njega nije lako odlučiti. Zbog toga je na tržištu sve više tzv *strimer traka* koje i same predstavljaju medij spoljne memorije — na strimer traku se relativno brzo i potpuno automatski prebacuje kompletan sadržaj hard diska ili na vaš zahtev samo datoteke koje su u bližoj prošlosti menjane

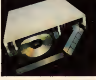
Laserski diskovi

Iako smo ovo poglavlje nazvali *spoljna memorija* često se koristi i izraz *masovna memorija*. Smatramo da je prva varijanta preciznija — odgovor na pitanje šta je masovno a šta nije se menja iz godine u godinu tako da je kapacitet flopi disk jedinica (npr 320 K ili 1,2 M) sada manji od radne memorije mnogih personalnih računara, nije nemoguće da će slična sudbina uskoro zadesiti i hard diskove. Laserske diskove i CD ROM-ove međutim mirne duše možemo nazvati masovnom memorijom — radi se o medijima koji obezbeđuju pamćenje do skora nezamislivih gigabajta i terabajta podataka!

Osnovna mana CD ROM-ova je što ni jedan „smrtnik" ne može da upisuje podatke na njih — od proizvođača treba naručiti veću seriju kompakt diskova da bi se proizvodnja uopšte isplatila. Zato su rezultati izvanredni — u okviru nedavno okončanog projekta poznata TV kompanija BBC proizvela je dva CD ROM-a koja sadrže detaljne mape čitave Velike Britanije — korisnik samo odlučuje kuda želi da krene a računar iscrtava mape slike važnijih objekata ulica kuća

Deo mana CD ROM-ova rešavaju takozvani WORM laserski diskovi. WORM je skraćenica od *Write Once Read Many times* — svaki korisnik koji se opremio odgovarajućom umereno jeftinom opremom može da upiše podatke na laserski disk ti podaci mogu da se čitaju proizvoljan broj puta ali je njihova promena nemoguća. Iako je kapacitet WORM diskova i dalje znatno manji od kapaciteta CD ROM-ova relativno jeftin medij masovne memorije od 4 gigabajta svakako izaziva poštovanje!

Za upotrebu CD ROM-ova i laserskih diskova potrebni su skupi drajvovi što znači da je ovakva masovna memorija i dalje van domašaja većine korisnika. Računarska vremena se, na sreću brzo menjaju!



Optički disk

Izlazne jedinice

U izlazne periferijske jedinice ubrajamo razne uređaje koji omogućavaju računaru da rezultate svoga rada na neki način saopšti korisniku. Periferijske jedinice bez koje se jednostavno ne može je monitor; gotovo isto toliko je potreban štampač, dok razni sintetizatori govora i dalje predstavljaju prilično egzotične naprave.

Monitori

Mnogi vlasnici kućnih računara i dalje posmatraju rezultate svoga rada na ekranima standardnih televizora. Ovakav je rad u mnogim slučajevima sasvim dovoljan pogotovu kada se uzme u obzir dubina naših džepova — kompjuterske igre amatersko programiranje i uopšte upoznavanje kompjutera zaista ne zahteva ništa bolje od televizora. Čim računar počne ozbiljno (tj profesionalno ili poluprofesionalno) da se primenjuje televizor postaje usko grlo pre svega kućni televizor obično služi da bi se gledala televizija. TV program nam, istini za volju odavno nije takav da bi kućni sporovi tipa *koji program da gledamo* bili česti ali je ipak u prosečnoj višečlanoj porodici televizor retko slobodan. Dugotrajan rad sa televizorom (naročito televizorom u boji) osim toga nije prijatan za oči a ni zdrav. Slika na televizoru je najzad nedovoljno kvalitetna za mnoge profesionalne primene kao što je obrada teksta (televizor ne može razgovetno da prikaže 80 znakova u svakom redu a to je neophodan uslov za komforno kompjutersko pisanje) ili naročito računarsko crtanje. Ostaje dakle, da se nabavi monitor

Monitor je unekoliko sličan televizoru njegov osnovni deo je katodna cev ili CRT. Ona je zasnovana na osobinama nekih vrsta fosfora koji svetle kada na njih usmerimo elektronski mlaz. Zamislimo jednu staklenu površinu prekrivenu fosforom na koju je upravljen top koji izbacuje elektronski mlaz. Sve tačke u koje ovaj mlaz pogodi će svetleti — pomerajmo top tako da skenira čitav ekran i sve će tačke na njemu biti sjajne. Ukoliko se top pomera dovoljno brzo naše oko neće biti u stanju da registruje gašenje pojedinih tačaka pa će nam se činiti da svetli čitav ekran!



Multisinhroni monitor

Da bi obišao sve tačke, elektronski top se kreće po cik-cak liniji: pošavši od gornjeg levog ugla, top iscrtava 313 linija i stiže u donji desni ugao, zatim sledi povratak sličnom putanjom i ispisivanje 312 linije koje se nalaze između prvih 313 čime je postupak završen (američki TV standard zvani NTSC je našao drugačiji što izaziva nekompatibilnost i mnogobrojne probleme).

Rekli smo da naše oko pokazuje tromost koja je učinila da „pokretne slike" postanu moguće: svaki vizuelni utisak traje bar četrdesetak milisekundi bez obzira što je objektivno možda kraći. Zato se proces prebrisavanja ekrana ponavlja u svakoj pedesetinki sekunde. Tražene frekvencije je naravno prevelika da bi uopšte moglo da se razmišlja o topu koji bi se fizički pokretao — umesto toga elektronski mlaz koji on emituje biva skrenut primenom magnetnog polja koje proizvode ugrađeni elektromagneti.

Pretpostavimo da je neko stavio kartončić kvadratnog oblika između topa i površine ekrana. Kvadratić će onemogućiti elektronskom mlazu da pogodi centralni deo ekrana koji će tako ostati zatamnjen dok će okolina biti osvetljena: korisnik će dakle videti tamni kvadrat na svetloj pozadini. Ako sada zamislimo da našem topu nekako naređujemo kada da emituje mlaz a kada da ga isključi, postaje nam jasno da je na ekranu moguće dobiti neku statičnu sliku komplikovaniju od običnog kvadrata. Menjajmo tu sliku dvadesetak puta u sekundi i eto crtanog filma.

Dodajmo sada malo boje našim razmišljanjima. Pretpostavimo da je ekran premazan trima vrstama fosfora od kojih jedna, kada je pogodimo elektronskim mlazom, pocrveni, druga pozeleni a treća poplavi. Dodajmo zatim našoj „artiljeriji" još dva „topa" koji će u punoj sinhronizaciji gađati tačke ekrana osvetljavajući tako svaku od njih u crvenu, plavu ili zelenu boju ili neku njihovu kombinaciju. Kao što će vam rado objasniti svaki slikar, kombinovanjem crvene, zelene i plave boje mogu da se dobiju sve ostale boje a sa njima i kolor televizor odnosno monitor.

Recimo, potpunosti radi, da postoje i specijalni monitori kod kojih elektronski mlaz ne skanira čitav ekran već, krećući se poput olovke, iscrtava samo osvetljene tačke, čime se dobija izvanredna statična slika i vrlo slaba animacija (čak ni brzina od četrdesetak hiljada kilometara na čas nije dovoljno velika za „pokretne slike" dobijene na ovaj način).

Ulaz u televizor

U prethodnom izlaganju smo ispustili jednu veoma važnu stvar — kako da saopštimo mlazu koju će tačku osvetliti a koju neće. Kretanje mlaza elektrona je jasno strogo periodično pa u svakom trenutku vremena treba na neki način saopštavati koji od topova „puca" a koji se „odmara". Potrebne su nam, dakle, tri ulaza koja odgovaraju crvenoj, zelenoj i plavoj boji a samim tim i tri žice koje će voditi od računara do monitora. Ovakav ulaz u monitor se zbog engleskih imena za tri pobrojane boje, naziva RGB ulaz i pruža potencijale za najbolju moguću sliku. Osim R, G i B signala potrebno je obezbediti sinhronizaciju (*sync* impuls na shemama) kojom se nećemo opširnije baviti — za prosečnog korisnika računara je sasvim dovoljno da zapamti da RGB monitori omogućavaju izvanredno čistu sliku visoke rezolucije u koloru.

TV aparati su na žalost i danas najčešće lišeni RGB ulaza. Razlog nije u tome što je ovako nešto tehnički neizvodljivo (čitaoci ovih redova koji su verzirani u elektroniku i TV tehniku znaju da je u svakom televizoru moguće pronaći R, G i B ulaze mada ovakvo traženje može da bude otežano ili u praksi onemogućeno

Periferijska oprema u „Računarima"

Računari su se u svojim redovnim brojevima mnogo puta bavili periferijskom opremom, detaljno opisujući princip delovanja svakog uređaja ili predstavljajući konkretne modele štampača, plotera, diskova, džojstika... Verujući da će mnogi čitaoci ovog kratkog osvrta poželeti da prošire znanje koje su stekli, ostatak ovog poglavlja posvećujemo svojevrsnom indeksu tekstova o periferijskoj opremi — tekstovi su sortirani po temama i, u okviru svake teme, po broju časopisa i strani: 3/15 znači da je tekst na 15. strani „Računara 3". Stare brojeve (pod uslovom da već nisu rasprodati) možete da naručite uz pomoć kupona koji ćete naći u ovom izdanju.

Masovna memorija: Disk jedinice (3/15), Disketna veza (4/24), Strogo kontrolisani diskovi (5/37), BBC B i diskovi (5/38), Sporo ali dostižno (C-64 i diskovi) (5/39), Strujama i kanalima po svetu (5/40), Prema svecu i drajv (Mikrodrajv) (7/56), S diskom ili na njemu (10/16), Hard diskovi (11/10), Neupisivo upisivanje (11/15), Optičke memorije (16/10)

Periferijska oprema: Vrata u svet (1/23), Radost crtanja (5/17), Kako će govoriti računari (5/35), Monitori (6/17), Svetlosna pera (7/17), Svet na dlanu (modemi) (8/17), Palice za igru (9/18), Video digitajzer (11/16), Igračke za odrasle (ploter) (15/12), Mišja groznica (21/18), S-s-s-samplujemo na spektrumu (22/18), Tajna crvena kutije (25/18), Računar koji čita (Digitizer) (31/14), Za svaku karticu (multisync monitori) (35/12).

Štampači: Štampači u akciji (2/32), Ćirilica na epsonu (3/62), Naki novi Epsoni (LQ-1500) (5/20), Canon protiv Epsona (PW-1080A) (7/7), Laserska veza (9/20), Matrični štampači (15/8), Upotreba Epson kompatibilnih štampača (16/27), Dvanaest veličanstvenih (štampača) (17/14), YU slova na Staru i LX80 (17/48), Čudo u štampanju (Panasonic 1092) (18/10), Mlaz koji misli (18/18), Štampači sa lepezom (19/8), Miran mlaz (24/13), Epson to radi bolje (24-pinski štampači) (28/10), Laser na pisaćem stolu (30/18), Ekonomska klasa (štampači) ide u raj (32/16), Kvantni skok štampača u boji (33/17), NLQ bez tajni (34/34), NLQ nad Zagrebom (LX-86 YU set) (36/47)

kod aparata koji koriste previše integrisanih kola) već u tome što do skora nije bilo načina da se RGB ulaz upotrebi — televizor je jednostavno igrao ulogu televizora!

Iako nemaju RGB, mnogi televizori imaju takozvani kompozitni video ulaz. Umesto tri žice do monitora dolazi samo jedna (ne računajući masu) kroz koju stižu informacije o bojama pojedinih tačaka kao i potrebni sinhronizacioni impulsi. Kako se jednom žicom mogu zameniti četiri? Grubo poređenje bi bila situacija u kojoj pošta montira dvojnika na vaš telefon. Na taj način dva korisnika mogu da se služe istom linijom ali se pojavljuje i ozbiljno ograničenje — ne možete da koristite telefon dok vaš dvojnik razgovara. Kada biste se međutim sinhronizovali sa dvojnikom tako da svako od vas govori u tačno određene sate i kada se ne bi pojavljivale situacije u kojima morate da iskoristite telefon kada nije na vas red, ni ovaj nedostatak se ne bi primećivao! Slično tome, nekoliko signala mogu da se modulišu na jedan signal-nosilac i tako upravljaju elektronskim topovima.

Jasno je da se kompozitni video signal mora „raspakovati" kako bi CRT dobila informacije neophodne za normalan rad. Transformaciju kompozitnog video signala u R, G i B signale obavljaju kola ugrađena u monitor pri čemu ta transformacija ne može da proizvede signale koji bi bili potpuno identični sa R, G i B signalima koji su „spakovani" u kompozitni video. Pojavile su se dakle izobličenja koje ugrožavaju kvalitet slike.

Kompozitni video signal je, na žalost, krajnje nepogodan za radio difuziju — da se ljudi nisu dosetili nekih trikova, gledali bismo isključivo kablovsku TV što se nikako ne bi isplatilo vlasnicima nepriključenih aparata. Da bi se slika slala kroz etar, treba kompozitni video signal RF modulisati. Antena na krovu vaše zgrade prima RF signal, on se jednom žicom (druga je masa) dovodi do antenskog ulaza u vaš televizor gde najpre prolazi kroz tjuner i (de)modulator, a zatim biva dalje transformisan u R, G i B signale koji jedini mogu da kontrolišu iscrtavanje slike.

Računar, jasno, nema nikakve potrebe da šalje sliku kroz atmosferu, on je sa monitorom vezan kablom. Da bi se obezbedila kompatibilnost sa starijim TV uređajima, u mnoge računare su ugrađeni RF modulatori koje ćete lako prepoznati — smešteni su u malu metalnu kutijicu negde uz samu zadnju stranu kućišta. Ovi modulatori transformišu kompozitni video signal u oblik koji bi antena primila iz atmosfere. Čim tako dobijeni signal stigne u TV prijemnik, on biva pretvoran u kompozitni video — otprilike kao kada biste razgovarali sa komšijom tako što biste ono što želite da kažete prevodili na engleski a zatim njemu prepuštali da tekst vrati na srpskohrvatski. Ma koliko dobri prevodioci radili ovakvo nešto, prevedeni tekst neće biti identičan polaznom. Slično tome, upotreba antenskog ulaza u televizor unosi izobličenje slike i onemogućava rad sa većim brojem karaktera u redu i grafikom visoke rezolucije.

Kako izabrati monitor

Bilo je teško ali se nadamo se isplatilo — sada zar ne znate kako radi monitor pa je došlo vreme da izaberete onaj koji vam odgovara. Treba pre svega da se odlučite za kolor ili crno-beli monitor što je, verujte nam na reč, prilično teška odluka. Rekao bi se da će onaj ko nema dovoljno novca kupiti crno-beli monitor a srećnik koji se nalazi u manjoj nestašici dolara i maraka — monitor u boji! Ne mora međutim da bude baš tako — crno-beli (češće crno-zeleni ili narandžasti) monitor obavezno daje kvalitetniju sliku iako je daleko jeftiniji. Razlog leži u činjenici da je crno-belom monitoru potreban samo jedan delić fosfora za svaku tačku dok kolor monitori zahtevaju tri, po jedan u svakoj boji (priđite kolor televizoru i lupom posmatrajte neku statičnu sliku — uočićete višebojnu strukturu svake tačke). Na ekranu umereno visoke cene obično ima oko 360 000 fosfornih tačaka što znači da crno-beli monitor može da „opsluži" rezoluciju 600•600, a kolor monitor sa istim brojem tačaka — svega 350•350! Za korisnike koji će se baviti obradom teksta, poslovnom obradom podataka i sličnim radnjama koje zahtevaju dugotrajno i koncentrisano gledanje u ekran crno-beli (ili, kako se to obično kaže, monohrom) monitor predstavlja pravi izbor — nije ni malo slučajno što se na sve terminale velikih kompjuterskih sistema ugrađuju ovakvi uređaji. Jeste li, međutim, nekada probali da igrate neku igru na velikom sistemu? Ako i niste, nemate za čim da žalite — igre bez boje kod kojih se po ekranu kreću slova i nisu nešto naročito. Kolor monitor bi, dakle, mogao da bude pogodan za igre mada je za njih i običan televizor više nego dovoljan. Sve u svemu, za naš džep je monohrom monitor upotpunjen kućnim televizorom u boji verovatno najprihvatljiviji izbor.

Multisinhroni monitori

Videli smo da nije lako izabrati monitor koji istovremeno odgovara karakteristikama našeg kompjutera i karakteristikama našeg džepa! Dodatnu nevolju pričinjava činjenica da kupovina novog kompjutera (pa čak i nove grafičke kartice za stari kompjuter) često znači da stari i ne baš jeftini monitor treba preneti na „suvo i mračno mesto" ili ga prodati za delić originalne cene. Zato su multisinhroni monitori poslednjih meseci pravi hit — jednom izdvojite nekih 600 funti i

nabavite monitor koji će zadovoljiti potrebe svih sadašnjih i mnogih budućih personalnih računara i video kartica!

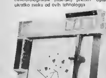
Govoreći o monitorima do sada smo pomenuli nekoliko brojki rekli smo na primer da se slika iscrtava dvadesetak puta u sekundi da mlaz ispisuje 312 5 linija u svakom prolazu i tako dalje — radi se o karakteristikama PAL odnosno NTSC sistema Da bi se vertikalna rezolucija računara povećala mora se povećati broj linija a samim tim i brzina pokretanja snopa elektrona koji za isto vreme treba da icrta više linija Brzinu kojom se snop kreće po svakoj od linija nazivamo *frekvencija horizontalnog skaniranja* dok je brzina kojom zrak putuje niz ekran *frekvencija vertikalnog skaniranja* Na većini postojećih monitora iole prihvatljivih cena *frekvencija horizontalnog skaniranja* je 15—35 KHz a *frekvencija vertikalnog skaniranja* 50—70 Hz — više frekvencije jasno odgovaraju boljim (i skupljim) monitorima

Kao što im ime i govori multisinhroni monitori automatski podešavaju frekvenciju horizontalnog odnosno vertikalnog skaniranja u zavisnosti od karakteristika primljenog signala Prilagođenje raznim računarima je uz to zahtevalo ugradnju raznih RGB ulaza koji su obično dopunjeni takozvanom linijom intenziteta *(intensity line)* koja obezbeđuje 16 (=2**4) boja Još kvalitetniju sliku nudi takozvani RrGgBb ulaz umesto četiri do računara vodi osam linija pri čemu postoje dva intenziteta za svaku boju pa čak i dva intenziteta za sam intenzitet ukupno 8 linija odnosno 64 (=2**8) boja

Svaki multisinhroni monitor može da radi i u analognom modu pri čemu svaka linija određuje intenzitet odgovarajuće boje Ukupan broj boja je praktično neograničen ali je kvalitet slike nešto slabiji U poslednje vreme pojavili su se i prvi multisinhroni monohrom monitori koji se pokazuju vrlo pogodnim za poslovne primene

Printeri i ploteri

Nama sumnja da je štampač najpotrebnija periferijska jedinica — on nam omogućava da rezultate svoga rada prenesemo na papir koji je i pored svog napretka računara i dalje glavni medij za razmenu informacija Izbor štampača nije ništa jednostavniji od izbora računara treba najpre da se opredelimo za vrstu štampača a onda da izaberemo jedan od tržišno raspoloživih modela Što se vrste štampača tiče na tržištu se nalaze matrični *daisy wheel* termički ink-jet i elektrofotografski (laserski) printeri Opišimo ukratko svaku od ovih tehnologija



Ploter

Matrični štampači

Matrični štampač proizvodi otisak udarajući iglicama u mastiljavu traku postavljenu između glave i papira Slova se dakle sastoje od tačkica pa je njihova čitljivost neprijatno slaba što je računarima donelo lošu reputaciju — mnoge štamparije i dalje odbijaju da slažu tekst ispisan na matričnom štampaču! Tačkasta struktura će se jasno manje primećivati ako su tačke gušće raspoređene što znači da štampači koji imaju više iglica nude kvalitetniji otisak Trenutno su se afirmisala dva standarda koje nazivamo 9 pinski i 24-pinski Ukoliko vam se skok sa 9 na 24 učinio prevelikim unekoliko ste u pravu glava 24-pinskog štampača zaista ima 24 iglice ali su one raspoređene u dve vaoma bliske kolone od po 12 iglica Desna kolona je malo smaknuta u odnosu na levu što znači da su iglice raspoređene u cik-cak Rastojanje između pina 1 i pina 24 je jednako rastojanju između prvog i devetog pina na starim modelima što znači da je veličina slova jednaka bitno su povećane rezolucija kvalitet otiska i na žalost cena — 500—600 funti za 80-kolonske i 1000 funti za 132-kolonske printere



Matrični štampači

Matrični štampači su u globalu umereno jeftini (250—350 funti) umereno brzi umereno jednostavni za održavanje i upotrebu i (zbog svega toga) veoma široko rasprostranjeni U raznim prospektima pročitaćete da matrični štampač u svakom sekundu ispisuje 100 120 ili čak 200 znakova ali su ovi podaci veoma varljivi — nikome nije jasno kako proizvođači do njih dolaze! Kvalitetan matrični štampač će u sekundi proizvesti svega šezdesetak slova što znači da će štampanje standardne šlajfne teksta (1960 znakova) potrajati otprilike pola minuta Korisniku je ostavljena mogućnost da uspori ispis na nekih petnaestak znakova u sekundi poboljšavajući otisak — svaki se red ispisuje po dva puta uz mikroskopsko pomeranje papira pa se stiče utisak da je matrica tačaka koja određuje svaki znak bitno povećana (npr 24•16) Na ovaj način nastaje takozvani NLQ ili *Near Letter Quality* ispis

Što se papira tiče svaki matrični štampač može da piše na odvojenim listovima koji se kreću oko pokretnog valjka Ovakav je papir ipak krajnje nepraktičan videli smo da je štampaču potrebno pola minuta ili minut da ispiše jedan list što znači da ćete posle svakog minuta štampanja morati da ubacujete novi papir posle izvesnog vremena utvrdićete da ubacivanje papira odnosi više vremena nego samo štampanje pa ćete potražiti racionalnije rešenje Najčešće se koristi perforirani kompjuterski papir (u stranoj literaturi *fan-fold paper)* koji se obično isporučuje u sanducima od kojih svaki sadrži po 2000 međusobno povezanih listova proširenih perforacijom sa svake strane Štampač koji treba da prima ovakav papir mora da bude opremljen takozvanim *traktorom* koji je obično uključen u standardnu opremu Osim papira repromaterijal predstavlja i traka koja se obično nabavlja u inostranstvu

Jedna od karakteristike koje su učinile matrične štampače popularnima je grafika — i računarska slika se sastoji od tačkica što znači da je možete preneti na papir! Rezultati ovakvih primena nisu na žalost baš idealni — veće zatamnjene površine nisu ravnomerne traka se prebrzo troši a glava sa iglicama postepeno propada

Printeri sa lepezom

Štampač sa lepezom funkcioniše na način unekoliko sličan običnoj pisaćoj mašini mali čekić udara u slovo koje se pozicioniralo na najvišu tačku pokretne lepeze dovodeći to slovo u dodir sa trakom koja na papiru ostavlja potpuno formiran otisak Slova se dakle ne sastoje od tačkica i u tome leži tajna izvanrednog otiska koji nude štampači sa lepezom Iz principa rada proizilazi jasno i velika mana ovakvih uređaja mogu se ispisivati samo slova koja su *unapred* ugrađena na lepezi što znači da je definisanje karaktera veoma komplikovano a grafika isključena Stranim korisnicima nemogućnost definisanja znakova ne mora smetati — za svega dvadesetak funti dokupljuju se lepeze sa specijalnim slovima koje se po potrebi umeću umesto standardne Obzirom da naša slova č ć ž i š ne postoje ni na jednoj poznatoj lepezi nemogućnost definisanja znakova za mnoge predstavlja dovoljan razlog za diskvalifikaciju štampača sa lepezom

Izvanredan utisak nije plaćen samo nepostojanjem grafike štampači sa lepezom su mnogostruko sporiji od matričnih pa ispisuju najviše 15—20 znakova u sekundu ukoliko dakle odlučite da ispišete rukopis od tridesetak strana (nešto poput teksta koji upravo čitate) računajte sa tim da ćete pored stola sa printerom provesti gotovo čitav sat! Dežurstvo pored štampača je



Štampač sa lepezom

naophodno zbog umetanja listova savršen otisak će biti još savršeniji kada se nađe na bank-postu ili nekom drugom dobrom papiru koji se obično kupuje u risovima. Potencijalnog kupca štampača sa lepezom treba upozoriti i na problem traka: dok se traka na pisaćoj mašini i matričnom štampaču bezbroj puta okrene da bismo je, kada izbledi, jednostavno zamenili novom „Jetraset" traka štampača sa lepezom prolazi ispred glave samo jednom posle čega je treba zameniti novom. Reklo bi se, dakle, da štampači sa lepezom imaju mnogo mana a malo vrlina, ali ipak ima dosta korisnika koje kvalitet matričnog štampača jednostavno ne može da zadovolji — takvi korisnici moraju da biraju između lepeze i (značajno skupljeg) lasera.

Laserski štampači

Elektrofotografski (laserski) štampači su zbog izvanrednog otiska i velike brzine rada sve popularniji kako u svetu tako i kod nas. Elektrofotografski štampač je zapravo mašina za suvo foto-kopiranje kojoj je dodat svetlosni izvor koji računar može da kontroliše (tipično laser). Uprošćeno rečeno, laser naelektriše foto-konduktivni doboš a zatim ovo naelektrisanje raspoređuje suvi toner (specijalni prah) po papiru koji najzad prolazi kroz fazu fiksiranja. Rezultati su izvanredni — rezolucija od 300*300 tačaka po inču se gotovo sasvim ravnopravno nosi sa foto-slogom. Ubedljiva demonstracija laserskog štampača su mnoge kompjuterske knjige koje se poslednjih meseci izdaju — ček i pažljivi pregled uz korišćenje snažne lupe neće otkriti tačkastu strukturu znakova osim možda kod kurzivnih slova. Dobra strana laserskih štampača je i što obezbeđuju mešanje teksta i grafike na istoj stranici, ovo mešanje olakšavaju razni komercijalni programi za takozvano stono izdavaštvo. Iako su do nedavno predstavljali skup raritet, laserski štampači su danes neophodan deo opreme svakog korisnika koji se profesionalno bavi obradom teksta — nekada basnoslovno visoke cene probile su granicu od 2000 dolara.

Ostale tehnologije

Termički štampači proizvode otisak delovanjem ugrejanih iglica na specijalni papir. Na nekim novijim modelima između (običnog) papira i glava postavlja se voštana traka — kada se traka sa zadnje strane zagreje, specijalni vosak se topi i ostaje na papiru. Termički štampači su jeftini i tihi ali proizvode otisak slabog kvaliteta trošeći pri tome specijalni papir koji se teško nabavlja. Kod nas je određenu popularnost postigao jedino Sinklerov štampač koji je, međutim, sasvim neprimeren bilo kakvom profesionalnom radu.

Ink-Jet štampači zasnivaju se na gađanju standardnog papira mlazom mastila — u svakoj se sekundi ispali nekih 50 000 kapljica! Preciznost se postiže bilo upotrebom piezoelektričnih kristala bilo minijaturnim grejačima koji nateruju mastilo u komorama da provri. *Ink jet* štampači su brzi, izuzetno tihi i troše veoma malo energije pa se često baterijski napajaju i prodaju u kompletima sa džepnim računarima. Nešto su skuplji od termičkih ali je održavanje veoma jeftino — koristi se običan papir dok se umesto traka kupuju glave sa ugrađenim rezervoarima mastila. Otisak je ipak nezadovoljavajući za poslovnu korespondenciju.

Laserski štampač



Kolor štampači

Projektna dokumentacija, pisama, knjiga i izveštaji uvek se izrađuju u crno-beloj tehnici. Napredak tehnologije je međutim, afirmisao tehnike kolor štampanja: plotere sa perima, elektrostatičke plotere, termalne, ink jet, matrična i elektrofotografska (laserska) štampače. Najpopularniji su **ploteri sa perima**, koji po papiru pišu uz pomoć specijalnog flomastera, pri čemu je softveru koji upravlja ploterom omogućeno da naredi zamenu tog pera flomasterom druge boje. Ploteri sa perima su u poslednje vreme umereno jeftini i masovno se koriste u biroima za projektovanje pri izradi planova, ne treba zaboraviti ni naučno-inženjerske primene u kojima je potrebna grafička prezentacija rezultata. Kvalitet dokumenta koji proizvodi solidan ploter sa perima je izvanredan jer su linije kontinualne a ne stepeničaste — rezolucija je praktično beskonačna, tj. ograničena jedino debljinom korišćenog pera. Broj pera koja mogu da se razmenjuju pod softverskom kontrolom je obično između 2 i 10, dok je maksimalna preciznost vrhunskih plotera oko 0.001 inč (3 stota dela milimetra) što znači da će linije čije je međusobno rastojanje veće od ovog limita zaista biti razdvojene. Ploteri sa perima se izrađuju u raznim verzijama i, zavisno od kvaliteta, koštaju 2—10 000 dolara.

Iako je ova tehnologija dotarana do savršenstva, njene mane nisu mogle da se uklone: ploter može da crta samo linije što znači da je rad sa njim spor, ispisivanje slova naporno a ravnomerno bojenje površina praktično nemoguće. Zbog toga se ukazala potreba za grafičkim uređajima koji bi sliku proizvodili na bazi rasterske grafike. Najkvalitetniji (i najskuplji) su **elektrostatički ploteri**: specijalni dielektrični papir prolazi ispod glave čija je širina jednaka maksimalnom formatu lista i koja se sastoji od velikog broja minijaturnih iglica. Neke od ovih iglica su dovedene na određeni potencijal što naelektriše papir. Papir zatim putuje iznad tečnog tonera, pojedini segmenti papira, u zavisnosti od svog naelektrisanja, privlače odgovarajuće količine tonera i tako nastaje jedna od boja. Postupak se ponavlja tri ili četiri puta sa različitim tonerima i tako nastaje izuzetno kvalitetna (obično 400 tačaka po inču) slika u boji.

Elektrostatički ploteri su relativno spori i izuzetno skupi: za ploter koji proizvodi papire formata A3 treba odvojiti 12 do 14 hiljada dolara dok A0 ploteri mogu koštati i čitavih 100 000 dolara, održavanje je uz to prilično skupo a papir treba posebno nabavljati. Ovakvi se uređaji, dakle, koriste samo tamo gde je potreban izu-



Kolor štampač

zetan kvalitet — Lucasfilm na primer koristi isključivo elektrostatičke plotere na kojima je proizveden dobar deo „pozadinskih prizora" iz „Ratova zvezda".

Termički štampači su mnogo jeftiniji: za nekih 300 dolara možete da nabavite kolor termički štampač sa pokretnom glavom. Postoje, međutim, i stranično orijentisani termički štampači u boji koji koštaju između 4 i 10 hiljada dolara i nude sasvim pristojan otisak.

Matrični kolor štampači, zapravo ne predstavljaju novitet: sećate li se starih dobrih pisaćih mašina koje su obezbeđivale dvobojni tekst? Čitavom dužinom mastiljava traka je podeljena na dve pruge: gornja je crna a donja crvena. Kada čekić na koji je utisnuto slovo udari u gornju polovinu trake, na papiru će se pojaviti crno, a kada mehanizam malo izdigne traku tako da čekić udari u njenu donju polovinu — crveno slovo. Na potpuno istom principu deluju i kolor matrični štampači: traka je podeljena na tri sekcije, što znači da se slika formira u tri prolaza — u svakom se prolazu iscrtava po jedna boja. Iako su kolor matrični štampači u priličnoj upotrebi kako zbog niske cene tako i zbog jeftinog održavanja, njihove loše strane nisu zanemarljive. Traka se pre svega neravnomerno troši što znači da će posle kraćeg vremena neke boje biti mnogo isteknutije od drugih. Veće površine koje obojita, zatim, nisu dovoljno ravnomerne i sjajne, ovakvo bojenje uz to polako ali sigurno upropašćuje glavu. Tačke, najzad, teško mogu da budu manje od 0,18 milimetara.

Elektrofotografski (laserski) kolor štampači u ovom trenutku predstavljaju pravo svetsko čudo: prvi modeli će se na otvorenom tržištu pojaviti početkom 1988. i koštaju 20 do 30 hiljada dolara. Rezolucija će biti 300—400 tačaka po inču a brzina neverovatnih desetak stranica u minuti. Laserski kolor štampač nastaje malom modifikacijom mašine za kolor fotokopiranje.

Vodeći svetski proizvođači štampača su dakle u potpunosti osvojili razne tehnologije kolor štampanja. Cene su, istini za volju, i dalje relativno visoke, ali ne sumnjamo da će u bliskoj budućnosti kvalitetni kolor štampači koštati manje od magičnih 2000 dolara — laserski printeri su, sećamo se, počeli masovno da se prodaju i koriste tek kada su probili ovu „magičnu granicu". Neko će možda reći da štampanje u boji nije naročito potrebno — standardi uglavnom zahtevaju crno-bele izveštaje. Istina je, međutim, nešto drugačija: standardi prate život, što znači da će izvanredna primena boja biti „pronađena" tek kada proizvodnja i umnožavanje kolor dokumentacije postane tehnički lako izvodljiv zadatak!

Sinteza glasa

Kada je pre pet godina Zoran Modli pripremio nekoliko reklama koje je „izgovarao" računar, veći deo auditorijuma je bio zabezeknut — zar tako jeftin računar kao što je „komodor 64" može da govori i to na našem jeziku? Sada svi znamo da može, ali i da jeftini računari u celini govore prilično slabo — ako unapred znate šta će biti rečeno, savršeno ćete razumeti, u protivnom...

Za pravu sintezu govora potreban je poseban kontroler koji nije naročito skup — nekih tridesetak funti. Ovakvi čipovi obezbeđuju više nego razgovetan govor čiji se parametri (npr. visina ili boja tona) po želji podešavaju. Potreban je, jasno, i odgovarajući program koji je obično prilagođen engleskom jeziku: za sintezu glasa koji bi govorio srpskohrvatski treba znati mnogo toga o strukturi jezika i uopšte fenomena. To, naravno, ne sprečava mnoge amatere da se igraju sa komercijalnim programima za „komodor 64" ili „atari ST" i postižu rezultate koji zadovoljavaju njihova želje.

Šta smo izostavili

U poslednjoj fazi pripreme ovog tematskog izdanja bili smo prinuđeni da zbog ograničenog prostora odustanemo od nekoliko najavljenih tema — kataloga engleskog obrazovnog softvera, upotrebe računara u školama van obrazovnog procesa i malih bejzik podsetnika za „spektrum", „komodor", „amstrad" koji nedopunjuju „osnovnu školu bejzika". Ove teme su potpuno pripremljene i biće objavljene u majskom i junskom broju „Računara".

3

Osnovna škola bejzika

Prošlo je već više od šesnaest godina otkako sam kao učenica trećeg razreda srednje škole, načinila prve korake u programiranju učeći o algoritmizaciji zadatka, ali i danas se živo sećam koliko je to mojim drugarima i meni bilo teško. Koliko je teško bilo mojim profesorima, koji su morali da predaju po tada unikatnom programu informatike bez udžbenika i literature, shvatila sam tek kada sam i sama postala profesor programiranja. Dosta neprospavanih noći, straha od pitanja na koja neću umeti da dam odgovor, pregledanja pismenih zadataka u kojima učenici nude „očigledna" rešenja i testiranja stotina kilobajta sopstvenih programa koji ne rade već su prošlost. Zato je otkrivanje tajni i mogućnosti novog školskog računara „tim 011" predstavljalo za mene izazov i zadovoljstvo, kako i treba da bude u prvim susretima sa računarom.

Međutim, mnogi od čitalaca koji tek treba po prvi put da se upuste u otkrivanje računara, uz radoznalost verovatno osećaju i strah. Za bojazan ima dosta osnova. Priručnik koji se dobija uz bilo koji računar, pa i školski, nudi sažetu informaciju koja je za početnike teško razumljiva. Udžbenik s druge strane, mora da prezentira nastavne sadržaje tako da se mogu praktično proveriti na bilo kom od računara vrlo raznolikih mogućnosti. Tako početnik, u stvari, nema odgovarajući prateći materijal koji bi mu pomogao da brzo i efikasno upozna mogućnosti konkretnog računara. Kada se tome doda da kod nas, i pored uvođenja računarstva u opšte obrazovanje, još uvek nije razrađena metodika predavanja programiranja, dovodi se u pitanje kako će i da li će praktične vežbe na računaru uopšte biti realizovane.

Zato će tekst „Osnovna škola bejzika", koji je nastao kao rezultat mojih priprema za praktične vežbe iz programiranja na računaru „tim 011", možda pomoći nastavnicima i učenicima da lakše izlože i savlađuju gradivo iz informatike. Imajući u vidu da većina korisnika računara nisu matematičari, objašnjevala sam i pojmove za koje se podrazumeva da ih programeri znaju. Verujem da niz konkretnih primera, kroz koje se ispituju karakteristike i mogućnosti bejzika, može značajno pomoći ne samo onima koji rade sa „timom", već svima koji tek treba da počnu da koriste računar. Praksa je pokazala da metod izlaganja kojim se učenici vode da postepeno sami otkrivaju tajne svog računara, razvija interes za programiranje i podstiče kreativno razmišljanje. Pokušala sam da korišćenjem ovog metoda izlaganja informacija o programiranju u bejziku pokažem i kako treba učiti programiranje.

Tekst je podeljen u osam poglavlja. Po mojoj proceni, za praktičnu realizaciju svakog sem poslednjeg od njih potrebno je po četiri školska časa. Razume se, u zavisnosti od predznanja učenika moguće je iste sadržaje savlađivati i za manje ili više vremena. Jedino treba stalno imati na umu da gradivo iz programiranja nije savladano onda kada ga nastavnik izloži, već onda kada su učenici u stanju da ga primenjuju.

OSAM BLISKIH SUSRETA

Nevenka Spalević

Prvi susret

Pokreni me. . .

Počinjemo druženje s računarom kratkom pričom o tome šta se dešava kada ga povežemo i uključimo. Na većini kućnih računara odmah po uključivanju pojavi se neka poruka koja se završava sa „ready“. Tom porukom se, u stvari, prijavljuje da je bejzik interpretator već u operativnoj memoriji i da odmah možete početi programiranje.

ROM I RAM



Interpretator nam je na raspolaganju odmah po uključivanju računara zato što je upisan u ROM — postojanu memoriju koja se tako zove jer se u nju ne može upisivati, može se jedino čitati njen sadržaj (Read Only Memory). Korisniku je za upis i čitanje na raspolaganju RAM operativna memorija. Sadržaj RAM-a gubi se po prestanku napajanja, pa će korisnik sve programe koji će mu trebati u daljem radu, a nisu u ROMu, morati da sačuva na nekom spoljašnjem memorijskom medijumu — kaseti ili disketi na primer. No šta je u stvari interpretator?

Pozivanje bejzika

Interpretator je sistemski program na mašinskom jeziku koji omogućava interaktivan rad na računaru i izvršavanje bejzik programa. Sam interpretator zauzima relativno velik prostor u memoriji, a koliko mesta preostaje korisniku za njegove programe i podatke može se videti iz početne poruke. No računar može da radi i bez bejzik interpretatora. Onaj minimum sistemskog softvera koji je neophodan za rad računara zove se operativni sistem. Operativni sistem, ili bar jedan njegov deo, takođe su ugrađeni u ROM. Bejzik interpretator se u svom radu oslanja na module operativnog sistema. I ne samo on. Ako radite na Paskalu, treba vam odgovarajući program prevodilac — paskal kompajler. Bejzik interpretator u tom slučaju samo smeta, njegovo mesto u memoriji može zauzeti kompajler koji će svoje funkcije realizovati takođe oslanjajući se na operativni sistem.

Stoga je rešenje koje nudi računar „tim 011“ bolje, jer se ne insistira na jednoj verziji bejzika. Samim uključivanjem računara ovde nećete dobiti nikakvu poruku. Operativni sistem i ostali softver nisu u ROM-u, već ih treba prethodno učitati sa sistemske diskete. Ako je ona na svom mestu u disk-jedinici, po uključivanju računara automatski će biti učitan operativni sistem. Oznaka A> iza koje trepće kursor govori da je sistem spreman da prihvati vaše zahteve.

Da biste učitali bejzik interpretator otkucajte GBASIC. Šta će se desiti? Ništa, jer računar nije ni uzeo ovu poruku u razmatranje. On pristupa analizi i izvršavanju zahteva tek po pritisku na dirku RET (return) koja za njega ima smisao tačke u rečenicama prirodnih jezika. Pritisak na bilo koju tipku, naime, proizvodi upis aski koda odgovarajućeg znaka u jedan deo memorije koji se zove bafer tastature. Onog momenta kada se u bafer unese znak čiji je aski kod jednak 13, sve dotad uneta informacija se uzimaju iz bafera i podvrgavaju analizi. Zato, ako neki primer ne radi šta bi trebalo, najpre proverite da li ste pritisnuli RET.

Ako ste ispravno pozvali bejzik interpretator na ekranu ćete dobiti informaciju o verziji. Podatak da raspolažete sa 21810 bajtova memorije i poruku OK, što odgovara onom READY kod većine kućnih računara. Ako ste pak napravili neku grešku, dobićete inverzno ispisanu poruku „TIM-011 DOS Editor & Error Handler“ i spisak komandi koje su nam na raspolaganju. Dirkom DELETE obrišite slovo po slovo pogrešne poruke, pa pritisnite RET da biste se vratili u sistem. Tada ispravno otkucajte GBASIC i možete nastaviti rad.

Rad s interpretatorom

Bejzik interpretator „tima“ razume više od 20 komandi. Za početak ćemo naučiti upotrebu onih koje se najčešće upotrebljavaju. To su:

LOAD za unošenje programa sa spoljašnje memorije diskete u operativnu,

RUN za startovanje programa koji je već u memoriji,

LIST za ispisivanje programa,

NEW za oslobođenje memorije pre unošenja novog programa i

SAVE za pamćenje programa iz memorije na disketi.

Osim komandi, interpretator je u stanju da izvršava i naredbe bejzika, pa računar možemo koristiti u neposrednom režimu rada kao moćan kalkulator. No da bismo na ekranu mogli da pratimo šta je računar uradio, upoznajmo odmah naredbu bejzika koju ćemo najčešće upotrebljavati — PRINT. Upravo zato što se često koristi, ova naredba ne mora da se piše slovo po slovo, umesto nje se može otkucati znak pitanja.

U daljem tekstu nećemo podsećati da na kraju poruke treba pritisnuti dirku RET, to se od sada podrazumeva. Otkucajte

```
? A (ili PRINT A)
Dobićete
 0
OK
Otkucajte sada
? "A"
Na ekranu ćete ugledati
A
OK
```

Naredba PRINT, koja služi izdavanju podataka na ekran, u prvom slučaju je odštampala vrednost promenljive A, a kako je prethodno nismo definisali, dobili smo njenu početnu vrednost koja je ovde 0. Napominjemo da nije tako kod svih računara („pekoma“ na primer). Obratite pažnju da nula nije štampana u prvoj koloni. Po konvenciji, ako je broj pozitivan znak + se ne piše, kao u našem slučaju, ali se ostavlja prazno mesto. Kada smo tražili da se štampa "A", dobili smo u prvoj koloni slovo A, jer sve što pišemo

između znakova navoda računar smatra znakovnom konstantom stringom i izdaja upravo ono što između navodnika stoji
Pokušajte sada se
PRINT 1 2 3
PRINT ,1", ,,2", ,,3"
U prvom slučaju dobijate
_1__2__3,
A u drugom
123

Oznaka (tačka i zarez) u listu PRINT naredbe (lista je sve ono što stoji iza službene reči PRINT) jeste znak razdvajanja Taj znak ukazuje interpretatoru da trebe štampati tri broja (1 2 i 3) a ne trocifreni broj 123 Primećujete da su između brojnih podataka ostavljena po dva prazna mesta a između znakovnih nijedno Ako pak otkucate
PRINT 1 −2
dobićete
_1_−2
Znači separator iza broja ostavlja jedno prazno zaštitno mesto da se brojevi ne bi mešali a druga praznina iz prethodnog primera popunjena je znakom broja
Proverite šta će se desiti ako sada u svim primerima tačku i zarez zamenite zarezom
Naredbe PRINT 1 2 3 će odštampati brojeve 1 2 i 3 sa znatno većim razmakom Precizno između svaka dva ostaviće po 14 praznih mesta
Naredbe PRINT ,1" , 2" ,3" će učiniti skoro isto, jedino će čitava štampa biti pomerena jednu kolonu udesno To na ekranu možete lepo uočiti ako ispod odgovora računara otkucate redom cifre od 1 do 0 tri puta na sledeći način
123456789012345678901234567890
?,,1" , ,,2 ,,3'
1________2__________3_
Tako ćete bez muke videti tačno pozicije na kojima su odštampani podaci

Znaci razdvajanja

O čemu se u stvari radi? Separator (zarez) štampa sledeći podatak iz liste na početku narednog polja po konvenciji Polje je prostor na koma se štampa podatak Dužina polja je broj kolona potrebnih za smeštanje podataka Za znakovne podatke dužina polja jednaka je broju znakova koji u njemu učestvuju a za brojne ima dve mesta više za znak broja i za zaštitni razmak na kraju broja Tako broj 1 zauzima tri kolone ali znak ,1" samo jednu Pošto numerički podaci mogu imati različit broj cifara za njih se po konvenciji ostavlja polje dužine 14 jer je to prostor na koma se može ispisati većina podataka Separator zarez poštuje ovu konvenciju i sledeći podatak ispisuje od početka prvog praznog polja, tj od 15 ili 29 kolone
Napravite sada sledeći eksperiment (eksperiment jer ovo ne piše ni u jednoj knjizi ali nas baš interesuje šta se dešava kada se uradi ono što nije objašnjeno)
PRINT 1 2 3
Dobija se
_1 2_ 3
Tačka koja je u prirodnim jezicima separator u programskim jezicima znači nešto sasvim drugo U američkoj notaciji za zapis broja njena je funkcija da razdvaja ceo od razlomljenog dela broja a kako otuda potiču računari ova notacija je preneta i u programske jezike Računar je prve dve cifre razdvojene tačkom protumačio kao broj 1 2 i napisao ga na svoj način kad 1.2 No šta je sa trećim? Ispisao je I 3 Konvencija u programiranju je da se sve što je suvišno što se podrazumeva ne mora pisati jer se tako šteda i prostor i vreme Zato se pravi decimalni brojevi tj oni čiji je ceo deo nula mogu zapisati i bez te vodeće nula Na primer umesto da pišemo 0 3 možemo samo 3 U našoj naredbi PRINT tačka očito nije bila protumačena kao znak razdvajanja a nikakav drugi znak razdvajanja nismo napisali Izgleda da nam je interpretator oprostio i snašao se na neki način Proverimo da li će razumeti ovo
PRINT ,1" ,2" 3"
Dobijamo sledeći ispis
1_0_2_0_3
Dakle ako podaci u listu nisu razdvojeni separatorima, ili bejzik interpretator će smatrati da je između njih trebalo da stoji znak One dve nule u ispisu vode poreklo od tačaka između znakovnih podataka Interpretator je smatrao da smo od njega tražili da odštampa broj 0 0 koji se ekonomično može zapisati kao 0 ili još kraće samo kao tačka Već vidite da pri radu sa računarom nije važno šta vi mislite da ste tražili već šta ste tražili po kovenciji koja važi za vaš računar
Probajmo sada sa dvotačkom
PRINT 1 2 3
Dobijamo poruku



_I
Syntax error
Interpretator je razumeo da tražimo štampanje i odštampao je 1 zatim je naišao za dvotačku koja za njega predstavlja separator između dve instrukcije ali šta smo hteli od njega porukom 2 to zaista nije mogao da razume Zato nam je i javio da smo napravili sintaksnu tj formalnu grešku Bejzik interpretator nije u stanju da otkriva logičke greške u našim programima a na formalne reaguje upravo ovako se Syntax error
Na kraju našeg prvog eksperimentisanja da kažemo još da prevodioci nisu toliko tolerantni' kao interpretatori, kod njih su sintaksna pravila mnogo stroža Isto tako ni svi bejzik interpretatori neće reagovati istovetno kad timov Na vama je da dobro upoznate svoj računar uz pomoć odgovarajućeg priručnika ali pre svega kroz praktičan rad imajte u vidu da u priručnicima mnogo šta uopšte ne piše dosta stvari je nejasno a ponešto i netačno

Direktan i programski režim

Sva komunikacija koju smo dosada vodili s računarom bila je u takozvanom direktnom ili neposrednom režimu rada Međutim osnovni način korišćenja računara je programski To podrazumeva da najpre pišemo niz instrukcija koje računar treba da upamti a zatim od njega zahtevamo da ih automatski jednu za drugom izvrši Taj niz instrukcija čini program koji se čuva u zato izdvojenoj zoni operativne memorije — zoni bejzik programa Uz program se čuvaju i podaci koje on koristi Oblast memorije u kojoj se podaci čuvaju zove se zona promenljivih informacija po kojoj bejzik interpretator zna da li treba odmah da izvrši naš zahtev ili treba da ga upamti jeste broj programskog reda za koji ćemo nadalje koristiti skraćenicu bpr
Naime ukoliko je naš zahtev započet cifrom bejzik interpretator čitavu poruku tretira kao jedan programski red i smešta je u zonu programa očekujući od vas nove zahteve Na primer ako napišemo
10 PRINT A
na ekranu se ništa neće odštampati
Ukoliko pak zahtev nije počeo brojem kao na primer PRINT A odmah će biti analiziran i ako nema sintaksnih grešaka izvršen
Kako se radi u programskom režimu ilustrovaćemo na sledećem primeru Želimo recimo da sastavimo program koji će računati procente U odeljenju imamo 32 učenika, od kojih je 5 odličnih 10 vrlodobrih 15 dobrih, 5 dovoljnih i 2 imaju slabe ocene Ako želimo da njihov broj izrazimo u procentima pomoćiće nam program u koji je ugrađen sledeći postupak
5 32=x 100 => x=100·5/32
Sam program mogao bi ovako da glasi
10 B=32
20 INPUT A
30 P=100·A/B
40 PRINT P
Kada ga unesete otkucajte komandu RUN i on će početi da se izvršava Prvo će se pojaviti znak ? što znači da računar od vas očekuje neki podatak. Recimo da smo uneli broj 5 Računar će odštampati 15 625 što znači da 5 predstavlja 15 625% od 32 Ako ponovo startujete program sa RUN i unesete podatak 32 dobićete da 32 čini 100% od 32
Ovo i nije neki problem koji bi se morao programirati ali je zgodan za ilustraciju kako se program kreira ispravlja razvija i testira Usput će nam pomoći da naučimo još neke naredbe bejzika

Grešiti je programerski

Velika je verovatnoća da ćete napraviti neke greške pišući program Ako to ipak niste učinili, pogrešite namerno U liniji 10 umesto INPUT A otkucajte IMPUTAA Posle RUN umesto znaka ? dobićete poruku
Syntax error in 10
OK
10
Pošto nije mogao da razume vaš zahtev u liniji 10 bejzik interpretator vas je prebacio u takozvani edit mod režim ispravljanja grešaka i očekuje da prepravite tekst linije 10 tako da može da ga razume Najprostije se možete izvući iz ove neprilike tako što ćete pritisnuti RET a onda iznova otkucati liniju 10 Primetićete da je ovom prilikom RET proizveo štampanje sadržaja linije 10 Međutim ovom načinu ima smisla pribeći samo kada ne znamo da koristimo komandu EDIT ili je sadržaj odgovarajućeg programskog reda kratak U svakom drugom slučaju bolje je editovati liniju Ukoliko niste u edit modu pređite u njega otkucavši EDIT 10 Pritisnite nekoliko puta razmaknicu spejs (space — u daljem tekstu <SP>) najveću tipku na dnu tastature Prvih sedam puta dobićete po jedno slovo na ekranu a nadalje računar ignoriše vaše pokušaje Probajte sada istu stvar sa (BS) bek spejs dirkom koja je u gornjem desnom uglu tastature Svaki pritisak briše po jedno slovo U edit modu funkcija ove dve tipke je da vas vode unapred i unazad kroz tekst odgovarajuće linije onako kako je on upamćen u memoriji Vodite računa da tekst na ekranu ne mora biti istovetan onom upamćenom u zoni programa
Komanda edit ima više potkomandi od kojih ćemo zasada naučiti tri umetanje < insert > brisanje < dilit — delete> i zamenu znakova < čejndž' — change>)
Najpre ćemo slovo M u tekstu naše komande zameniti slovom N To radimo ovako



Ako isprva teško ide ne posustajte budite uporni Korišćenje linijskog editora teško će pasti samo onima koji su navikli na ekranski editor poput onog na „komodoru" koji omogućava da se jednostavnim dovođenjem kursora na odgovarajuću poziciju obavi zamena umetanje ili brisanje Ali šta možemo koristimo ono što imamo — pravo bogatstvo komandi i naredbi koje naš interpretator može da izvršava

Format LOAD Program [,R]

Ovde kao i ubuduće, podrazumevamo da tekst između malih uglastih zagrada < > treba zameniti nečim konkretnim a da je ono što stoji između srednjih zagrada [ ] opcija tj nešto što možemo a ne moramo koristiti Specijalno pri učitavanju programa moramo posle LOAD navesti ime programa između znakova navoda na primer LOAD "PROBA" će učitati program "Proba" sa diskete Naš interpretator razlikuje

se učitava i izvršava s tim što opcija R ostavlja datoteke otvorenima

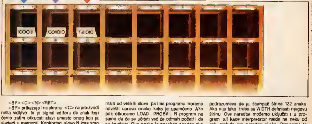
Komanda RUN automatski izvršava CLEAR, tj briše sve promenljive informacije sa staka i zatvara otvorene datoteke Ako želimo da ovo izbegnemo koristimo komandu CONT koju ćemo upoznati kasnije ili naredbu GOTO

Program se izvršava sve dok interpretator ne naiđe na naredbe kraja (STOP ili END) ili dođe do poslednjeg programskog reda ili naiđe na sintaksnu grešku Na poslednji način program se završava neregularno No u svakom slučaju interpretator se vraća u direktan režim rada

Ispisivanje programa

Format LIST [broj reda 1 [[broj reda 2>]]]
LIST broj reda 1] – [<broj reda 2]]]

Ove komande ispisuju listaju program prva na monitoru, druge na štampaču Pri tom se



<SP><C><N><RET>

<SP> prikazuje i na ekranu <C> ne proizvodi ništa vidljivo to je signal editoru da znak koji ćemo zatim otkucati stavi umesto onog koji je sledeći u memoriji Konkretno slovo N koje smo potom otkucali zamenilo je slovo M koje je tu bilo ranije <RET> Proizvodi štampanje daljeg neizmenjenog dela linije

Komandom LIST 10 Proverite šta je sada u liniji 10 Računar javlja

10 INPUTAA

Nastavljamo sa EDIT 10 Sledeća akcija je izbacivanje suvišnog slova A To radimo ovako

<SP><SP><SP><SP><SP><D><RET>

Na ekranu najpre dobijamo tekst INPUT, pritisak na <D> proizvodi pojavu slova A između kosih crta što znači da je taj znak izbačen iz memorije (iako ga vidimo na ekranu) a <RET> završava editovanje ostavljajući preostali deo linija neizmenjen Ponovo sa LIST 10 proveravamo šta imamo trenutno u memoriji i dobićemo 10 INPUTA Preostalo je još da umetnemo jedno prazno mesto između INPUT i A To posle EDIT 10 radimo ovako

<SP><SP><SP><SP><SP><I><SP><RET>

Pritisak na tipku I nije proizveo vidljiv efekat ekranu to je bio znak editoru da u programsku liniju umetne niz znakova koji će zatim biti pritisnuti

Međutim, ovako sporo ispravljanje grešaka bilo bi veliko mučenje i većina korisnika bi brže iznova otkucala čitavu liniju Srećom sve ove aktivnosti mogu se uraditi jednim postupkom zahvaljujući dirki (ESC) (iskejp — escape) u gornjem levom uglu tastature koja ukida dejstvo prethodne potkomande edit Tako bi celokupna ispravka IMPUTAA u INPUT A tekla ovako

<SP><C><N><SP><SP><SP><D><I>
<SP><ESC><SP><RET>

Ne sumnjamo da ćete u praktičnom radu imati dovoljno prilike da uvežbate ovu komandu

mala od velikih slova pa ime programa moramo navesti upravo onako kako je upamćeno Ako pak otkucamo LOAD "PROBA", R program ne samo da će se učitati već će odmah početi i da se izvršava. Ova opcija je posebno zgodna ako komandu LOAD koristimo u programskom režimu rada Tako se naime može povezati rad više programa

Izvršenjem komande LOAD gubimo program i sve promenljive koji su prethodno bili u memoriji i automatski zatvaramo sve datoteke Međutim uz opciju R neće se zatvoriti datoteke

Neka nam sledeći postupak posluži kao primer

Programu za računanje procenata možemo dopisati liniju

1 REM PROBA

i snimati ga sa SAVE "PROBA" Zatim otkucajmo NEW čime ga brišemo iz memorije Zatim otkucajmo LIST Ako smo dobro radili na ekranu se ništa neće pojaviti jer nema šta da se lista Pošto smo se uverili da je memorija tj zona bejzik programa prazna otkucamo

LOAD "PROBA", R

Šta će se desiti? Dobićemo znak pitanja kojim program traži da se unese broj za koji će odštampati koliko čini procenata od 32

Startovanje programa

Format RUN [broj reda
RUN ime programa [,R]

Prvi format važi za programe koji su već u memoriji Ako se ne navede broj reda program se izvršava od početka u suprotnom od naznačenog broja reda Probajmo na sledećem primeru

10 A=10
20 PRINT A

Posle RUN biće odštampan broj 10 ali posle RUN 20 biće odštampana nula

Drugi format važi za programe koji nisu u memoriji U tom slučaju odgovarajući program

podrazumeva da je štampač širine 132 znaka Ako nije tako treba sa WIDTH definisati njegovu širinu Ove naredbe možemo uključiti i u program ali kada interpretator naiđe na neku od njih prekida sve naredne akcije Proverimo to sledećim programom

10 PRINT „+++++"
20 REM
30 LIST
40 PRINT "!!!!"

Ovaj program će po startovanju ispisati oluseve zatim izlistati samog sebe i stati Linija 40 koja treba da odštampa uzvičnike neće se izvršiti Pokušaj da nastavimo izvršavanje programa sa CONT propada program se ponovo samo izlista.

Brisanje programa u memoriji

Format NEW

Komanda NEW briše trenutno aktuelan program u memoriji i sve promenljive Teoretski može biti korišćena i kao sastavni deo programa ali moramo biti svesni da će izbrisati sve što je trenutno u memoriji Istina ova komanda ne briše program fizički nego menja pointere — pokazivače u kojima se čuvaju informacija o početku i kraju programa i podataka. Aktuelni program se ne menja i može se pronaći promenom odgovarajućih pokazivača ali ovo je bez naredbe OLD koja „oživljava" program vrlo težak zadatak

Ako se ne izbriše stari program u memoriji može doći do nekontrolisanog mešanja dva programa i njihovih promenljivih

Pamćenja programa

Format SAVE ime programa> [,A/ P]

Ovom komandom se program iz memorije snima na disketu pod imenom koje mu damo Dodatak A znači da će program biti snimljen u takozvanom aski formatu što omogućava da ga

učita neki tekst procesor (na sistemskim disketama je i program WS — WORD STAR) i radi s njim kao s ma kojim tekstom. Isto tako, ovaj format pamćenja programa omogućava i primenu korisne komande MERGE kojom se povezuju programi iz memorije sa ranije upamćenim programima. Stoga je naša preporuka da se svi programi pamte na ovaj način.

Dodatak P (protected) snima program u zaštićenom obliku. Kada se kasnije pozove (sa LOAD ili RUN) moći će samo da se izvršava, ali ne i da se lista ili ispravlja. Vrlo zgodna stvar za nastavnike koji žele da propituju učenike pomoću odgovarajućih programa, ali savetujemo da na svojoj disketi čuvaju originalnu verziju u tekst obliku.

I još jedan praktičan savet. Neka učenici ne rade sa originalnim sistemskim disketama. Jeftinije je kupiti disketu više nego rizikovati da se upropasti operativni sistem.



Drugi susret

Promenljive bez tajni

Svaki Programski jezik gradi se nad skupom osnovnih simbola koji čine azbuku jezika. Od njih se, poput reči i rečenica u prirodnim jezicima, grade elementarne i složene konstrukcije programskog jezika. Mogućnost i namena jezika vide se već prilikom definisanja njegovih elementarnih konstrukcija: konstanti, promenljivih, nizova i izraza. Kako su u našem bejziku rešene konstante i promenljive?

Konstante i promenljive

Ovaj deo treba da objasni pojmove „konstanta" i „promenljiva" i zato krenimo od značenja ovih termina. Latinski termin konstanta označava neku nepromenljivu vrednost. Primenjeno na brojeve to može da se objasni ovako. U jednačini

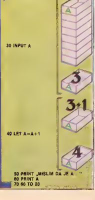
O=PI*D

PI je konstanta. Njena vrednost se nikad ne menja, bez obzira na to gde se koristi. Tu se pojavljuju još dva slova: O i D. To su oznake za obim i prečnik kruga koji mogu da se menjaju. Naime, za veći krug O i D će biti veći nego O i D nekog manjeg kruga, a PI je uvek isto.

Brojevi 2, 0 i −5 takođe su konstante ali konstante koje se u jednoj bitnoj stvari razlikuju od PI. To su celobrojne konstante, koje programeri zovu „intidžeri" (integers) i oni se tačno mogu predstaviti u računaru. Setimo se brojne prave koju smo upoznali još u nižim razredima osnovne škole. Celi brojevi bili su raspoređeni po njoj kao perle na ogrlici. Ali uskoro smo naučili da to nije čitava priča o brojevima jer ih ima beskonačno mnogo. Na žalost, na računaru možemo da radimo samo sa konačnim veličinama. Da bismo utvrdili kako računar pamti cele brojeve, otkucajmo sledeći program

```
10 INPUT A
20 PRINT A
30 GOTO 10
```

Startujmo program i unesimo neki trocifreni broj, 313 na primer. Broj će biti odštampan u



istoj formi. Probajmo sada sa sedmocifrenim brojem 3131313 na primer. Dobićemo 3.13131E+06. Ako imamo u vidu da oznaka E+06 znači šesti stepen broja deset, vidimo da je računar vaš broj 3 131 313 zaokružio na 3131 310, a ona tri je naprosto izgubio. Još gora stvar se dešava ako radimo sa milijardama. Računar u tom slučaju zaokružuje na hiljade. I što je najgore, sa veoma velikim celim brojevima ne može uopšte da radi, a i oni su nam često potrebni.

U vezi sa pamćenjem brojeva ima još problema, jer između svaka dva cela broja postoji takođe beskonačno mnogo brojeva. Možda vam se čini da su sve ovo opšte poznate stvari, ali ako želite da shvatite kako računar radi i koliko može, moramo se zadržati i na njima. Više godina radim kao profesor programiranja sa đacima koji su odlični matematičari, ali ne bih se mogla zakleti da je većini jasno zašto računari ne računaju tačno. A osnovni problem je upravo ovde, u načinu registrovanja podataka.

Dakle, suzimo problem na interval između brojeva 0 i 1. 1/2 se nalazi tačno na sredini. Probajte šta ćete dobiti ako kao ulaznu veličinu u naš program unesete 1/2. Računar odbija da odgovori i javlja

?Redo from start

Radi se o tome da računari ne pamte racionalne brojeve kao razlomke, nego ih beleže u decimalnoj formi. Međutim, umeju da dele, pa izmenite liniju 10 u

10 A=1/2

a liniju 30 izbacite i startujte program. Sada ćete dobiti vrednost .5, što je tačno, jer i u matematici 1/2=0.5. Mnogi razlomci se mogu potpuno tačno pretvoriti u decimalne brojeve. Proverite na računaru sledeće primere

3/8=.375 17/34=.5 7/5=1.4

Ali šta se dešava sa našom prijateljicom trećinom? Trebalo bi da „tim" s lakoćom podeli jedan sa tri, ali ne uspeva. To ne uspeva čak ni najbolji računar na svetu zato što je decimalan

zapis broja 1/3 beskonačan. Ni jedan računar ga nikada tačno nije zapisao. U decimalnoj formi 1/3 izgleda otprilike ovako 0.333333333333333 33333 ... a kod „tima" .333333

Izmenite sada program na sledeći način:

```
10 A#=1/3
20 PRINT A#
```

Program će štampati da je 1/3=.3333333432672208!

Smisao oznake # na kraju imena promenljive je da sračuna njenu vrednost „dvostruko tačno". Mi smo i dobili 10 decimala više, ali netačnih!

Ako pokušate sa 1#/3 dobićete korektan rezultat. Još je rano da pričamo zašto je tako. Imajte u vidu jedino da računari mogu tačno da predstave samo malu količinu brojeva, za nešto više mogu da upamte približnu vrednost, a sa beskonačno mnogo uopšte ne mogu da operišu.

No računari se ne koriste isključivo za računanja. Oni su u stanju da obrađuju i tekstove, a tekstovi se sastoje od reči. Zato bejzik jezik uz numeričke može da operiše i sa znakovnim konstantama. Reči „cica", „maca" i ma koje druge niske simbola između navodnika za računar predstavljaju znakovne konstante. Kao što mala ženska torbica rešava problem čuvanja mnoštva sitnica, pa dame koje je nose treba da vode računa samo o jednoj stvari, torbici, tako i računar sve ono što se nalazi između znakova navoda tretira kao samo jedan — znakovni podatak, ne zalazeći u to iz čega se sastoji i šta znači.

Između navodnika možemo pisati sve znake — slova, cifre, razmake i sve ostale na ekranu vidljive i nevidljive simbole sa tastature. Jedino uključivanje znakova navoda u string, kako programeri često zovu znakovne podatke, može da bude problem, ali uskoro ćemo i to rešiti.

Ime treba nadenuti

Kada od računara zatražimo da sačuva neke informacije, on mora znati kako da ih prepozna. Drugim rečima, svakom podatku koji je potrebno ponovo koristiti treba da se dodeli ime po kome će ga računar prepoznavati i pronalaziti kad zatreba. Zamislimo memoriju kao niz poštanskih sandučića za stanare jednog solitera. Svakom od njih pridruženo je ime, pa kada želimo da upamtimo neki podatak, jednostavno ga smestimo u jedan od „sandučića" čije ime znamo. Jedino treba poštovati pravila imenovanja promenljivih.

Kod „tima" ime promenljive može imati najviše 40 simbola. Proverite ovo tvrđenje na sledećem primeru:

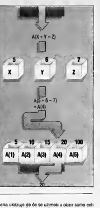
```
10 INPUT AAAAAAAAAAAAAAAAAAAAAAAAA
20 PRINT "TEST"
```

(U imenu treba da se pojavi 41 znak.)

Posle RUN dobijate znak pitanja, interpretator od vas traži podatak jer je naišao na naredbu INPUT, ali čim unesete neki broj, štampa Syntex error in 10, jer ne može da koristi takvo ime promenljive kakvo smo mu zadali.

Izbacite poslednje slovo iz imena u liniji 10, startujte program i on će sada korektno odštampati „TEST".

Kod „tima", kao i kod svakog drugog računara, ime promenljive ima da se sastoji samo od slova i cifara, pri čemu prvi simbol mora biti slovo. Kao ime se ne smeju koristiti rezervisane reči (THEN, SWAP i druge službene reči bejzika.) Kako najčešće radimo sa promenljivim kojima se mogu dodeliti vrednosti obične tačnosti, njihove imena se najjednostavnije pišu bez specijalnih dodataka. Ako želite da se promenljive računa dvostruko precizno, na kraju njenog imena stavićete znak #, na primer A#. Znak % na kraju



imena ukazuje da će se uzimati u obzir samo celi brojevi, a znak $ da se radi o znakovnoj promenljivoj.

Očistite memoriju od prethodnih eksperimenata sa NEW i otkucajte sledeći program:

```
10 A=1.23456789
20 B%=1.23456789
30 C#=1.23456789
40 PRINT A,B%,C#
```

Mada ste svim promenljivim dodelili istu vrednost, dobićete tri različita broja. Promenljive A ima vrednost 1.23457 jer se u običnoj preciznosti radi sa 6 značajnih cifara. Promenljiva B% biće prikazana kao 1 jer je to celi deo unetog broja, a promenljiva C# tačno onoliko koliko smo uneli 1.23456789, jer se u dvostrukoj preciznosti pamti čak 16 značajnih cifara.

Kada se govori o promenljivima treba razjasniti još dva pojma. To su tekuća vrednost i oblast definisanosti promenljive. Vrednost promenljive odgovara sadržaju memorijskog prostora koji je dodeljen promenljivoj sa odgovarajućim imenom. U našem primeru promenljiva B% ima vrednost 1. Oblast definisanosti je skup svih vrednosti koje bi mogle biti dodeljene promenljivoj, sve vrednosti koje mogu da „stanu" u odgovarajući memorijski prostor. Tako su celobrojnim promenljivim dodeljena dva bajta i to namenski, samo za ceo deo broja. Kada pokušamo da tu smestimo broj 1.23456789 vidimo da može da stane samo njegov ceo deo, razlomljen biva „odsečen". Ako pak pokušamo da na isto mesto stavimo broj 33000, to neće uspeti čak ni sa „odsecanjem", jer je broj prevelik. Ova situacija zove se prekoračenje i računar je prijavljuje sa Overflow. Međutim, ako promenljivoj A dodelimo broj 33000, to sasvim normalno ide jer broj 33000 pripada oblasti definisanosti promenljivih obične tačnosti.

Slično važi i za znakovne (azbučne) promenljive. Ako azbučnoj promenljivoj pokušate da dodelite bilo koju nisku do 255 znakova, neće biti problema. Ali pokušajte sa A$=1 ili sa dodelom nekog teksta dužeg od 255 znakova. U prvom slučaju dobićete poruku Type mismatch, a u drugom računar počinje da „pišti" jer od njega tražite nešto što ne može da uradi. Pokušajte da ga „prevarite" tako što ćete definisati tri azbučne promenljive A$, B$ i C$ od po 90 znakova pa tražiti da upamti promenljivu D$=A$+B$+C$. Nećete uspeti: interpretator vam poručuje String too long.

Registrovanje brojeva

Za one koje interesuje kako u stvari računar beleži brojeve evo nekoliko reči o tome. Prvi problem je kako u memoriji zabeležiti znak broja. Postoji nekoliko metoda predstavljanja označenih brojeva u računaru. Izbor konkretnog oblika registrovanja značajno utiče na način realizacije aritmetičkih operacija. U uobičajenoj matematičkoj notaciji svakom broju prethode znaci + ili − kojima je označuju pozitivni i negativni brojevi. Bilo bi praktičnije da se za znak broja ne uvode posebni kodovi, već da se on reprezentuje nekom od cifara. Tako je uvedena konvencija da 0 na mestu za znak broja reprezentuje pozitivne brojeve, a najveća cifra brojnog sistema (u binarnom je to 1) reprezentuje negativne brojeve. Ovakav zapis označenih brojeva zove se oblik znaka i apsolutne vrednosti.

Međutim, zbog dva različita zapisa nule i zbog komplikovanog vršenja aritmetičkih operacija u ranim računarima uveden je takozvani oblik nepotpunog komplementa u kome se kod negativnih brojeva svaka cifra broja zamenjuje svojom dopunom do najveće cifre brojnog sistema, a za zapis znaka broja važe ista pravila. Zapis u nepotpunom komplementu omogućava jednostavnije vršenje aritmetičkih operacija, ali i dalje ostaju dva različita zapisa nule.

Zato se u savremenim računarima za zapis brojeva najčešće koristi potpuni komplement koji se za negativne brojeve dobija tako što se na mestu najmanje težine zapisu u nepotpunom komplementu doda 1.

Opseg celih brojeva koji se mogu zabeležiti u registrima računara primenom navedenih zapisa zavisi od broja bitova koji su na raspolaganju. Tako se kod mikroračunara za zapis celih brojeva obično koriste jedan ili dva bajta, ali se, recimo kod familije računara VAX, može koristiti 5 različitih dužina reči za zapis celobrojnih podataka.

Primetimo da je interval brojeva koji se mogu predstaviti u potpunom komplementu nesimetričan u odnosu na 0: možemo zapisati jedan negativan broj više. Na primer, od 256 različitih zapisa koji se mogu naći u bajtu, jedan je rezervisan za zapis nule, 127 za pozitivna i 128 za negativna brojeve.

Kako se za zapis celobrojnih podataka u „timu" koriste dva bajta, to je oblast definisanosti celobrojnih promenljivih interval [−32768,32767]. I sada je jasno zašto je za broj 33000 prijavljivano da je prevelik.

Mešoviti brojevi, to jest brojevi koji imaju i ceo i razlomljen deo, beleže se u registrima računara na dva načina: u fiksnom i pokretnom zarezu. Metoda fiksnog zareza podrazumeva da je pozicija decimalne tačke unapred određena, te se ne mora zapisati eksplicitno, već se podrazumeva. U naučno-tehničkim primenama uobičajeno je da se ovaj način koristi samo za registrovanje celih brojeva, a metod pokretnog zareza za sve ostale. Međutim, u poslovnoj primeni nije tako. Računari predviđeni za komercijalnu obradu podataka rade najviše sa podacima zapisanim u fiksnom zarezu bez obzira na to da li su celi ili ne, a pokretni zarez se koristi samo u izuzetnim slučajevima.

Zapis broja X u pokretnom zarezu podrazumeva da se broj predstavlja u obliku para (Xm

Xe), tj. X=Xm*N↑Xe, gde je N osnova brojnog sistema u kome je zapisan X. Veličinu Xm zovemo mantisa broja, a veličinu Xe eksponent. Zahvaljujući ovakvom zapisu broja registrujemo samo njegove značajne cifre, te ovakav metod služi za registrovanje vrlo velikih i vrlo malih veličina koje izlaze iz opsega brojeva koji se mogu predstaviti u obliku fiksnog zareza (Značajne cifre su sve cifre različite od nule i nule između značajnih cifara. Na primer u brojevima 12000 i 0.00012 značajne su samo cifre 1 i 2, a u broju 102000 značajna je i nula između cifara 1 i 2.) Da bi zapis brojeva u računaru bio jednoznačan koristi se normalizovana mantisa (ona koja je pravi razlomak čija je prva cifra iza pozicione tačke različita od nule — u binarnom sistemu to mora biti 1, ali u dekadnom 9 različitih cifara mogu da se nađu na poziciji 10↑−1)

Za zapis mantise u pokretnom zarezu obično se izdvajaju tri do četiri bajta u kojima se ona piše u obliku znaka i apsolutne vrednosti. Broj cifara mantise utiče na tačnost registrovanja brojeva. Tako ako za nju izdvojimo tri bajta možemo računati na 6 značajnih dekadnih cifara, kad što je to slučaj kod „tima" u običnoj tačnosti. Ako za zapis mantise izdvojimo četiri bajta dobijamo 9 značajnih dekadnih cifara rezultata, što je slučaj kod „spektruma", „komodore" i većine drugih kućnih računara. Ako pak izdvojimo sedam bajtova za zapis cifara mantise kao što je to slučaj kod dvostruke preciznosti na „timu", imamo čak 16 značajnih dekadnih cifara.

Prilikom određivanja veličine eksponenta treba imati u vidu u kom sa brojnom sistemu zapisuje broj. Sam eksponent se doduše zapisuje u binarnom obliku, ali se u nekim rešenjima podrazumeva da se broj i mantisa beleže u heksadekadnom sistemu. Po određivanju eksponenta on se zapisuje u obliku potpunog komplementa uvećanog za 1 na poziciji znaka eksponenta (kôd „višak 128" kod nas i većine mikroračunara).

Zavirimo u memoriju

Napravićemo sada jedan malo složeniji eksperiment. Zavirićemo u „timovu" memoriju da vidimo kako on u stvari beleži promenljive. Da bismo razumeli sledeći program treba da objasnimo neke funkcije koje početnicima obično nisu bliske.

U programu se pojavljuje funkcija PEEK. Njen format je PEEK(<i>), gde <i> može imati vrednost od 0 do 65535. PEEK saopštava šta se nalazi na memorijskoj lokaciji čija je adresa <i>, a to saopštenje možemo učiniti i za nas vidljivim ako ispred funkcije stavimo PRINT.

Druga funkcija koju koristimo je VARPTR. Njen format je VARPTR (<ime promenljive>). VARPTR daje adresu prvog (najmlađeg) bajta promenljive. Pre poziva ove funkcije promenljivoj se mora dodeliti vrednost, u suprotnom interpretator poručuje „illegal function call", što znači da je funkcija pozvana na nedozvoljen način.

Možemo da počnemo sa istraživanjem. Otkucajte NEW i unesite sledeći program:

```
10 INPUT A%
20 ADR=VARPTR(A%)
30 FOR I=0 TO 1
40 PRINT PEEK (ADR+I);
50 NEXT I
60 PRINT
70 GOTO 10
```

Program očekuje da unesete neki ceo broj koji memoriše pod imenom A%. ADR je adresa najmlađeg bajta ove promenljive. Kako se celi brojevi registruju u dva bajta u ciklusu se štampaju samo sadržaji bajtova ADR i ADR+1. PRINT u liniji 60 treba da obezbedi da se znak pitanja kojim se od vas očekuje unos sledećeg podatka ne štampa u istom redu sa sadržajima bajtova što bi se inače desilo jer se iza naredbe PRINT u ciklusu nalazi zarez.

Startujte program i unesite najpre nekoliko brojeva manjih od 256, jer oni u binarnom zapisu imaju manje od 9 cifara pa mogu da stanu u jedan bajt. Za sve njih dobićete „sliku" koja u mlađem bajtu ima sam uneti broj, a u starijem 0. Probajte sada sa 256. Njegova „slika" je 0 1. Broj 512 biće prikazan kao 0 2. Znači, jedinica u starijem bajtu vredi 256 jedinica mlađeg bajta. Probajte i sa drugim brojevima većim od 255 i pokušajte da formulišete pravilo kako izgleda zapis pozitivnih celih brojeva.

Probajte sada sa −1. Dobićete 255 255. Binarno to bi bilo 11111111 11111111, odnosno potpuni komplement broja −1. Možete li sada da odgovorite na pitanje kako se registruju negativni celi brojevi?

Prekinite izvršavanje programa pritiskom na dirku (<BRK> (gore desno na tastaturi) i izmenite ga tako što ćete umesto A% u linijama 10 i 20 staviti A i u liniji 30 staviti da I ide do 3, jer „tim" promenljive u običnoj tačnosti beleži u 4 bajta.

Startujte program i unesite broj 1. Dobijate „sliku" 0 0 0 129.

Dekadna jedinica koju smo uneli kao vrednost promenljive A i u binarnom sistemu je 1, odnosno $0.1 \cdot 2^{1}$. Znači, u normalizovanoj mantisi sve cifre sem prve iza pozicione tačke su nule, a eksponent je 1. Kako se podrazumeva da je mantisa normalizovana, ta prva jedinica se ne piše. Zato su sva tri bajta u kojima se čuva mantisa nula. Eksponent se beleži sa „viškom 128" (radi lakšeg poređenja brojeva) pa je zato naš eksponent 1 zapisan kao 129.

Unesite sada brojeve 2, 4, 8, odnosno stepene broja dva i primetićete da se jedino eksponent menja. Probajte sada sa 15. Dobijate „sliku" 0 0 112 132.

Broj 15 u binarnom brojnom sistemu se piše 1111, odnosno $0.1111 \cdot 2^{4}$. Eksponent je dakle 4, odnosno uvećan za 128 iznosi 132, kao što i piše. Najstariji bit najstarijeg bajta mantise rezervisan je za znak broja koji je pozitivan, pa je u njemu 0. Zatim slede tri jedinice iz mantise (jer se prva iza pozicione tačke podrazumeva). Znači sadržaj trećeg bajta je 01110000, odnosno 64+32+16=112.

Probajte sada sa brojevima manjim od 1. Broj 0.125 je zgodan primer jer je njegov prevod u binarni brojni sistem konačan: $0.001 = 0.1 \cdot 2^{-2}$. Svi bajtovi njegove mantise biće nule, a eksponent 126, odnosno broj za dva manji od 128, što ćete i videti na ekranu. Preostaje vam da proverite kako izgleda zapis negativnih brojeva u pokretnom zarezu.

Pristupite još jednoj izmeni programa. U redovima 10 i 20 A zamenite sa A#, a ciklusna promenljiva I neka se kreće do 7. Testirajte program sa istim vrednostima kao za jednostruku tačnost, a pokušajte i sa brojevima koji imaju više od 6 značajnih cifara.

Na kraju iskoristite program za utvrđivanje kako „tim" beleži znakovne podatke. Pri tome treba da imate u vidu da VARPTR kod znakovnih promenljivih daje adresu bajta u kome se čuva dužina promenljive. Za njim u sledeća dva bajta možemo pročitati adresu memorijske lokacije na kojoj se čuva prvi znak promenljive.

Nizovi i izrazi

Pošto smo se tako malo bolje upoznali sa konstantama i promenljivim „timovog" bejzika, ostaje još da vidimo šta su nizovi i izrazi. Tako ćemo kompletirati priču o osnovnim gradivnim jedinicama jezika od kojih se prave složene konstrukcije — naredbe, programski redovi, potprogrami i programi.

Kada u matematici treba da radimo sa velikim brojem promenljivih, nije nam zgodno da za svaku pamtimo posebno ime, već im dajemo jedno zajedničko, a međusobno ih razlikujemo po indeksu. Na primer, ako imamo 32 učenika, za obradu njihovog uspeha mnogo nam je zgodnije da ih označimo sa $u_1, u_2, \ldots, u_{32}$ nego sa Aleksić, Babić ... Pri tome nam u programiranju ne odgovara da indekse pišemo kao što to činimo olovkom na papiru, jer smo u obavezi da i ime i indeks smestimo u istu liniju. Zato pišemo u(1), u(2), ..., u(32). Ako usput treba da napravimo i statistiku ocena po predmetima, znači ako nam treba svaka ocena svakog učenika, još zgodnije je da svakoj oceni pridružimo po dva indeksa. U tom slučaju o(i,j) predstavlja ocenu i-tog učenika iz j-tog predmeta. Na primer o(2,11) je ocena drugog učenika iz 11. predmeta. U mom odeljenju to je ocena Nenada Bogojevića iz programiranja i ona od početka školske godine ima vrednost 5.

Dakle, u programiranju je često potrebno raditi sa grupom podataka koji imaju zajedničko ime, a međusobno se razlikuju jedino po indeksu. Takva grupa podataka zove se niz. Broj indeksa koji pridružujemo svakom članu niza zove se dimenzija niza. Po broju dimenzija razlikujemo jednodimenzione, dvodimenzione i višedimenzione nizove. Jednodimenzionim nizovima iz programskih jezika u matematici odgovaraju nizovi, a dvodimenzionim — matrice. Kako učenici prvog razreda srednje škole još ne znaju šta su to matrice, možda je bolje ne opterećivati ih uopšte dvodimenzionim i višedimenzionim nizovima.

Na „timu" teoretski mogu da se koriste nizovi do 255 dimenzija, pri čemu je maksimalan broj elemenata po jednoj dimenziji 32767. Kažemo teoretski, jer je korisniku koji radi u bejziku na raspolaganju manje od 22K memorije. Pri tom za razliku od rada sa običnim promenljivim, rad sa nizovima koji imaju više od 10 članova zahteva prethodno rezervisanje memorijskog prostora. Da bismo objasnili o čemu se zapravo radi, napravimo analogiju sa uobičajenom životnom situacijom. Kada u hotel stigne nekoliko turista, za njihov smeštaj se obično nađe dovoljno prostora. Problemi nastaju ako odjednom treba smestiti stotinu nenajavljenih gostiju. Stoga se u ovakvim slučajevima unapred rezerviše mesto. Bejzik interpretator je izuzetno predusretljiv domaćin. Za mali broj promenljivih, čak i ako su u grupi, to jest čine niz, obično pronađe mesto u memoriji. Ali ako mislite da mu pošaljete mnogo podataka, morate prethodno rezervisati memorijski prostor naredbom DIM u kojoj ćete naznačiti broj dimenzija, broj članova za svaku dimenziju i vrstu niza. Ovo poslednje činite odgovarajućim nastavkom u imenu niza koje se formira kao ime promenljive. Sve ove informacije su za interpretatora izuzetno važne, jer nije svejedno da li treba da odvoji 100*2 ili 100*8 bajtova za članove nekog niza. Napominjemo da u ovoj verziji bejzika ne možemo, kao u nekim drugim programskim jezicima, koristiti negativne indekse.

Izrazi su poslednje elementarna konstrukcija koja za razliku od složenih sama za sebe ne može proizvesti akciju na računaru. Izrazi koje koristimo u bejziku mogu biti brojni, znakovni, relacijski i logički. U brojnim izrazima učestvuju brojni podaci. Rezultat izračunavanja brojnog izraza u regularnom slučaju takođe je broj koji se može registrovati u računaru. Pokušaj svakog neregularnog izračunavanja prijavljuje se odgovarajućom porukom o grešci.

U znakovnim izrazima učestvuju znakovni podaci. Nad njima se vrše znakovne operacije i funkcije, a kao rezultat opet dobijamo znakovni podatak. Međutim kod relacijskog izraza ono što bismo uslovno mogli nazvati „oblast definisanosti" i „oblast vrednosti" ne predstavlja isti skup kao što je to bilo kod brojnih i azbučnih izraza. Naime, u relacijskom izrazu mogu učestvovati

brojne ili znakovne vrednosti, ali rezultat njegovog izračunavanje može biti samo „tačno" ili „netačno". Kod logičkih izraza pak logičke operetore primenjujemo ne veličine tipa „tačno" „netačno" i dobijamo kao rezultat vrednost tog istog tipa. Svim vrstama izraza pozabevićemo se detaljno kasnije kada budemo upoznavali naredbe bejzika.

Treći susret

Kako namignuti

Naučili smo već da ispisujamo koji god želimo tekst na akranu pomoću naredbe PRINT. Sada ćemo obraditi ispisivanje takata na ma kojam maatu akrana i ispisivanje oaančenim ili trapćućim slovima. Usput ćemo upoznati nova komande bejzik interpretatora koje pomažu programarima pri krairanju i teatiranju programa, naatavljajući ujedno postapen razvoj našag prvog obrazovnog programa.

Ne praktičnim vežbame iz programirenje zapaža se de učenici izuzetno vole de se služe grafičkim mogućnostima računara. Dovoljno je da im se za nekoliko minuta objasni kako se crtaju tačka, linija i krug pa de u toku istog časa napišu programe za crtanje složenih likova. Inspirisani željom de što pre kreireju i ožive svoje likove, shvateju odmah dejstvo naredbi prenosa, upravljanja i korišćenja ciklusa. Gradivo za čije smo teoretsko izlaganje ranijih godina trošili nedelje, sade učenici uz praktičan red ne računaru sa dobrom grafikom savleduju za dan. Zato ćemo ovde predstaviti naredbe za crtanje na samom početku i iskoristiti ih za ilustraciju dejstva drugih naredbi bejzika.

Programske olakšice

Rekli smo već da je naš interpretator toleran-tan, nije formalist kao prevodioci koje ćemo kasnije upoznavati. Uz to je i vrlo ljubazan prema korisnicima, meda to možda niste mogli primetiti jer ste ga upoznavali preko editora. No pošto smo prvu prepreku, ispravljanje programa, već savladali, možemo da se upoznamo sa olakšicame u programiranju koje nudi naš interpretator.

Zapazili ste u gornjem redu tastature neoznačene dirke. To su takozvane funkcijske dirke čije se dejstvo može programski definisati i menjati. Pod kontrolom operetivnog sistema one imaju jednu ulogu, a pod kontrolom bejzik interpretatora njihovo dejstvo je drugačije. Možemo pročitati sa ekrana šta koja znači, ali je za rad udobnije da preko njih stavimo nalepnice na kojima je ispisano odgovarajuće djejstvo. Ako ih sleva na desno označimo sa F1, F2, ... F11. Pritiskom na svaku od njih možemo videti de imaju sledeće dejstvo:

F01 — LIST
F02 — RUN (RET)
F03 — LOAD
F04 — SAVE
F05 — CONT (RET)
F06 — FILES' •BAS
F07 — TRON (RET)
F08 — TROFF (RET)
F09 — CLS (RET)
F10 — EDIT
F11 — SYSTEM (RET)

Ove najčešće korišćene komande ne morate kucati slovo po slovo, dovoljno je pritisnuti odgovarajuću funkcijsku dirku.

• Primećujete da uz komande dodeljene tipkama F2, F5, F7, F8, F9 i F11 stoji oznaka (RET). To znači da rade kao de smo uz odgovarajuću komandu pritisnuli i (RET), odnosno automatski se izvršavaju. Posebno budite oprezni sa dirkom F11 jer vas izvodi iz bejzika i vraća u operativni sistem.

Dejstvo prve četiri komande, kao i EDIT, već smo upoznali u prvom susretu sa računarom. Komanda CONT nastavlja izvršavanje programa koji je bio zaustavljen naredbama STOP ili END ili pritiskom na (BRK). Program se nastavlja tačno od onog meste gde je bio prekinut. Dok je program zaustavljen, korisnik može u direktnom režimu rad proveriti ili izmeniti neke promenljive. U fazi testiranja programa naredbu STOP možemo umetnuti u sva kritična mesta u kojima želimo da kontrolišemo vrednosti promenljivih. Proverite to sledećim primerom:

```
10 PI=0  C=1
20 PI=PI+4/C-4/(C+2)
30 PRINT PI
40 C=C+4  GOTO 20
```

Pritisnite F2 ili otkucajte RUN. Pred vama će početi da promiče niz brojeva koji se približavaju vrednosti broja PI. Pritisnite
 i program će se zaustaviti uz poruku u kojoj liniji je stao. Otkucajte u neposrednom režimu ?C i dobićete informaciju do kojeg C se stiglo. Zatim pritisnite F5 ili otkucajte CONT. Program će nastaviti da se izvršava. Opet ga prekinite i proverite vrednost za C. Videćete da se program nastavio od vrednosti koju je imala promenljiva C u trenutku kada ste prethodni put prekinuli program.

U Pauzi možemo izlistati program i zatim sa CONT nastaviti njegovo izvršavanje. Međutim, ako na ma koji način pokušamo da editujemo program, ili ako progam stane zbog toga što je naišao na sintaksnu grešku, naredba CONT neće moći da nastavi njegovo izvršavanje. Umesto toga dobijamo poruku Can't continue.

Napominjemo da program za računanje broja PI koji smo naveli nije napisan prema najboljem algoritmu za rešavanje ovog problema. Izabrali smo ga zato što se na njemu lako može videti dejstvo komande CONT.

Pritisnite sada tipku F7 ili otkucajte komandu TRON. Ako zatim startujete program, ili nastavite njegovo izvršavanje sa COMT, dobićete ispred svake aproksimacije broja PI sledeći tekst [40] [30] [20], tj. biće ispisani svi brojevi naredbi koja su se izvršavale između dva štampanja. TRON je naime komanda koje na ekranu ispisuje brojeve programskih redova koji se izvršavaju. TROFF ime suprotno dejstvo i ukida važenje TRON.

Pritisak na tipku F6 ili kucanje FILS • BAS propraćeno sa <RET> izlistaće imena svih bejzik programa sa diskete koja je trenutno u disk jedinici. Pri tome se ne briše program koji je u memoriji.

CLS je prva od grafičkih naredbi koje upoznajemo. Njena je funkcija brisanje ekrana i možemo je koristiti kako u direktnom tako i u programskom režimu rada.

Moje prvo iskustvo

Vratimo se na naredbu PRINT, jer smo pokazali samo deo njenih mogućnosti. Ako želimo da odštampamo poruku u kojoj se neki znak ponavlje više puta, to možemo učiniti navođenjem tog znaka potreban broj puta između znakova navoda u listi PRINT naredbe. Na primer, ako hoćemo de neke poruke naglasimo uokvirivši ih zvezdicama, kao

```
**********
* MILENA *
**********
```

to možemo neracionalno rešiti na sledeći način:

```
10 CLS
20 PRINT "**********"
30 PRINT "* MILENA *"
40 PRINT "**********"
```

Međutim, ako umemo da koristimo funkciju STRING$, čitav posao možemo obaviti ovako:

```
10 CLS  ?STRING$ (10,42)  ? "* MILENA *"  ?STRING$ (10,42)
```

Funkcija STRING$ (<a>, <b>) naime daje reč koje se sastoji od <a> znakova koji imaju aski kod <b>. Dakle, da smo u našem programu zamenili 10 sa 80, umesto 10 zvezdica bilo bi odštampano 80. Vremenom ćete upamtiti vrednosti aski kodove najčešće korišćenih znakova, a pre nego što nastupi taj trenutak koristite funkcije ASC i CHR$.

Funkcija ASC ima format ASC (<string>). To je numerička funkcija koja proizvodi aski vrednost prvog karaktera iz niske <string>. Na primer, ASC („A") daje 65, a ASC („BRAT") daje 66.

Funkcija CHR$ ima format CHR$ (<brojni izraz>). Oznaka $ na kraju imena signalizira da je to znakovna funkcija. CHR$ proizvodi jedan krakter čija je aski vrednost jednaka argumentu funkcije. Argument mora biti iz intervala [0,255]. Ako je kao argument navedena vrednost u pokretnom zarezu, odbacuje se razlomljeni deo. Na primer, CHR$ (65) proizvodi A, CHR$ (34) proizvodi navodnika, CHR$ (13) proizvodi prelazak u sledeći red, a CHR$ (42) daje znak *.

Kao što vidite, funkcije ASC, CHR$ i STRING$ mogu doste da pomognu u formatiranju izlaznih izveštaja. Problem koji se nedavno činio teškim, kako odštampati neku poruku između znakova navoda, na primer

Matematičke gimnazije „Veljko Vlahović"

sada rešavamo u jednom redu:

? "Matematičke gimnazije „"CHR$ (34), "Veljko Vlahović", CHR$ (34)

Još bolje možemo kontrolisati ispis ako u svoj repertoar instrukcija uvedemo naredbu TEXT. Ona omogućava da ma koji tekst ispišemo ma gde na ekranu.

Pri običnom načinu rada možemo neki znak ispisati na 80•24=1 920 različitih mesta jer imamo 24 reda sa po 80 kolona. Međutim, „tim" omogućava da kontrolišemo osvetljavanje tačka na ukupno 512•256=131 072 pozicije, to je njegova rezolucija. Ako ekran zamislimo kao prvi kvadrant Dekartovog pravouglog koordinatnog sistema, onda tačka u njegovom donjem levom uglu predstavlja koordinatni početak, to jest ima koordinate (0,0). (Kod nekih drugih računara tačka (0,0) je u gornjem levom uglu ekrana.)

Svaki znak koji se štampa na ekranu zauzima prostor od 6•10 tačaka, znači da u okviru jednog reda možemo odštampati više od 80 znakova, preciznije 512/6=85,3, ali ne pomoću naredbe PRINT. Uverite se u to otkucavši

CLS TEXT STRING$ (85,65), 0,0

Vidite da je svih 85 slova A stalo u jedan red. Ako pak u ovoj liniji naredbu TEXT zamenite sa PRINT, u jednom redu biće odštampano samo 80 slova A, a preostalih 5 će biti u sledećem redu.

Slično, umesto u 24 reda, pomoću naredbe TEXT možemo pisati u 256/10=25,6 redova. Već to je dovoljan razlog da je bolje upoznamo.

Mogućnost naredbe TEXT

Naredba TEXT ima format

TEXT <niska znakova>,<x>,<y>[,<m>[,<l>]]

Posle službene reči TEXT možemo pisati azbučni podatak između znakova navoda ili neki azbučni izraz koji želimo da štampamo počev od tačke sa koordinatama (<x>, <y>). Naredna dva

parametra koje smo ovde označili sa (m) i (i) nisu obavezne. Peremetar (m) označava način crtanje koji može biti apsolutan ili relativan. Kod apsolutnog crtanja ide se baš na tačku (<x>, <y>). Ako se treći paramatar na navede podrazumeva se da je 0, što znači da apsolutno crtamo (u ovom slučaju crtamo tekst). Ako je treći parametar naveden i ako mu je vrednost 1, onda se koordinate tačke od koje crtamo dobijeju tako što se na prethodne koordinete grafičkog kursora dodaju vrednosti <x> i <y>, odnosno crtamo relativno. (Matematičar bi rekli da to odgovare translaciji koordinatnog početka na poziciju grafčkog kursora.) Možda se pitate šta je to grafički kursor? Pa to je stvar koja u grafičkom načinu rada odgovara trepćućem kvadratiću koji ukazuje na tekuću poziciju. Razlika je u tome što se grafički kursor ne vidi i što se može specijalnom naredbom MOVE postaviti na ma koju od 131.072 tačke ekrana.

Četvrti parametar <i> označava intenzitet ispisa. Na „timu" raspolažemo sa 4 intenzitete. Intenzitet 0 se ne vidi na tamnoj pozadini, intenzitet I je bled, intenzitet 2 nešto jači, a ispis intenziteta 3 se najjače ističe na tamnoj pozadini. To možete uočiti pomoću sledećeg programa.

```
10 CLS
20 TEXT „PROBA", 0, 0, 0, 0
30 TEXT „PROBA", 0, 20, 0, 1
40 TEXT „PROBA", 0, 40, 0, 2
50 TEXT „PROBA", 0, 60, 0, 3
```

Kakvo je dejstvo parmetra koji definiše da li će crtanje biti na apsolutnim ili relativnim koordinatama ilustruje ovaj program.

```
10 CLS
20 MOVE 200, 100
30 TEXT „PROBA", 0, 20, 0, 3
40 FOR I=1 TO 1000: NEXT
50 TEXT „PROBA", 0, 20, 1, 3
```

Zadatak naredbi u liniji 40 je da uspori izvršavanje programa. U nekim bejzicima u tu svrhu se koristi naredba PAUSE, ali kako je mi nemamo, naterali smo računar da broji „u sebi" do 500 da bismo lakše mogli da pratimo rad programa. Primećujete da naredbe iz redova 30 i 50 koje se razlikuju samo u trećem parametru proizvode ispis na dve različite mesta ekrana.

Umetanje petlje za pauzu i mogućnost ispisa teksta intenzitetom 0 omogućava nam da postignemo efekat treptanja teksta ili bilo koje slika. Umesto imena Milena iz programa koji sledi u svom eksperimentu stavite svoje ime ili bilo koji drugi tekst.

```
10 CLS
20 TEXT „MILENA", 250, 120, 0, 3
30 FOR I=1 TO 500: NEXT
40 TEXT „MILENA", 250, 120, 0,0
50 FOR I=1 TO 500: NEXT I
60 GOTO 20
```

Efekat programa je da ime koje ste ispisali trepće na sredini ekrana. Program morate prekinuti pritiskom na (BRK). Eksperimentišite sa promenom dužine trajanja pauze u linijama 30 i 50.

Novi ogled pokazaće kako možemo ispisivati zasenčena slova. Program glasi:

```
10 CLS
20 TEXT „MILENA", 250, 120, 0, 1
30 TEXT „MILENA", 249, 121, 0, 3
```

Eksperimentišite sa ispisom pomerenim udesno i nadole i drugačijim intenzitetima. Pokušajte da od prethodna dva programa napravite jedan u kome trepću zasenčena slova ili reč „diše" – to jest naizmenično se ispisuju i brišu reči sa razmaknutim i normalno ispisanim slovima.

Predviđeni kraj

Iskoristimo ovo što smo naučili za pisanje jednog malog programa koji ćemo kasnije koristiti kao sastavni deo mnogih drugih programa. Postoji naime komanda MERGE koja učitava prethodno upamćen program, a bez brisanja programa koji je trenutno u memoriji. MERGE ćemo ubuduće često koristiti jer praktično omogućava povezivanje više programa i tako značajno povećava efikasnost programera koji jednom razrađene rutine kasnije umeće u svoje programe.

Počećemo od kraja. Jedna od odlika profesionalno pisanih programa je da se ne završavaju na nepredviđen način, da ne „padaju". To se postiže pored ostalog i kontrolom ulaznih podataka, odnosno sprečavanjem korisnika da unese ulazne veličine s kojima program ne bi korektno radio. Kontroli ulaznih podataka i eventualnih grešaka u računanju posvetićemo kasnije dosta pažnje, a sada ćemo se pozabaviti regularnim, urednim završetkom programa.

Ispravan program se obično završava nailaskom na naredbu STOP ili END. U prvom slučaju prijavljuje Break in ..., a u drugom javlja samo OK, što je dovoljno ako smo ga pisali za sopstvene potrebe. No ako želimo da pišemo obrazovni ili neki drugi program koji će upotrebljavati drugi korisnici, moramo ga projektovati tako da i onima koji nisu programeri bude jasno kako treba da ga koriste. Uz to, važno je i da od početka do kraja bude „ljubazan" prema korisnicima. Zato ćemo napisati rutinu završetka rada koja najpre proverava da li korisnik zaista želi prekid ili ne, a zatim ispisuje trepćuću poruku ONDA, DOVIĐENJA! na sredini ekrana.

Program smo numerisali linijama od 900 nadalje jer pretpostavljamo da naši budući programi neće imati više od 100 linija. Tako će se po primeni MERGE „KRAJ", kako ćemo nazvati ovu rutinu, ona naći na završetku programa iz memorije.

```
900 CLS: TEXT "ZELITE LI DA PREKINETE S
RADOM (D/N)",150,120,0,3
910 O$= INKEY$
920 IF O$<>"D" AND O$<>"N" GOTO 910
930 IF O$="N" THEN RUN
935 FOR I=1 TO 5
940 CLS:TEXT "ONDA, DOVIDJENJA.",200,100
,0,1
950 TEXT "ONDA, DOVIDJENJA.",199,101,0,3
960 FOR J=1 TO 500: NEXT J: TEXT "ONDA,
DOVIDJENJA",200,100,0,0: TEXT "ONDA, DOV
IDJENJA.",199,101,0,0
970 NEXT I
```

Listing 1

U programu se pojavljuje funkcija INKEY$. Ona prihvata znak otkucan na tastaturi i dodeljuje ga promenljivoj O$. Linija 920 ne dozvoljava da se program nastavi sve dok korisnik ne pritisne dirku N ili D. Ako korisnik ne želi da prekine rad, program će biti vraćen na početak komandom RUN. U ovom trenutku, kada je rutina KRAJ sama u memoriji, taj početak je linija 900. Međutim, kada je nadovežemo na neki drugi program, RUN će učiniti da se program izvršava od samog početka.

Crta, crta, crtica

Stvari koje smo naučili u naredbi TEXT efektno možemo primeniti u naredbama za crtanje. Za početak ćemo posvetiti pažnju naredbi MOVE, koju smo već pominjali, i naredbi PLOT za crtanje tačke. U krajnjoj liniji ova poslednja nam je dovoljna za crtanje ma kojeg lika, što ćemo pokazati na nekoliko primera. Međutim, uskoro ćete videti da su nam na raspolaganju i naredbe za crtanje koje zamenjuju čitave programe pisane korišćenjem jedino naredbe PLOT.

Opšti oblik naredbe MOVE je

MOVE <x>, <y>[, <m>[, <i>]]

Značenje parametara istovetno kao u naredbi TEXT. Pri tome efekat naredba MOVE nije odmah vidljiv na ekranu. Vrednost intenziteta postavljena naredbom MOVE biće korišćena u svim sledećim vidljivim grafičkim naredbama, ukoliko se tamo drugačije ne definiše.

Naredba

PLOT <x>, <y>, [<m>[, <i>]]

crta tačku na poziciji (<x>, <y>) ako je parametar <m> jednak 0 ili izostavljen, a na poziciji <x> mesta udesno i <y> mesta iznad prethodne pozicije grafičkog kursora ako je treći parametar jednak 1. Intenzitet crtanja 0 praktično briše prethodno nacrtanu tačku, a ako se izostavi, tačka se crta sa prethodno uvedenim intenzitetom. Ako osvetlimo sve tačka sa koordinatom y jednakom 0, na ekranu dobijamo vidljivi deo x ose. Probajte to sledećim programom:

```
10 CLS
20 FOR X=0 TO 511
30 PLOT X,0
40 NEXT X
```

Slično možemo dobiti vidljivi deo y ose pomoću programa

```
10 CLS
20 FOR Y=0 TO 255
30 PLOT 0,Y
40 NEXT Y
```

Na odgovarajući način se mogu dobiti linije paralelne x ili y osi. Ovaj program na listingu 2 crta mrežu „kvadratića".

```
10 CLS
20 FOR Y=0 TO 255 STEP 10
30 FOR X=0 TO 511
40 PLOT X,Y
50 NEXT X
60 NEXT Y
70 FOR X=0 TO 511 STEP 10
80 FOR Y=0 TO 255
90 PLOT X,Y
100 NEXT Y
110 NEXT X
120 GOTO 120
```

Listing 2

Primećujete da ne dobijamo prave kvadratiće već mrežu pravougaonika, mada smo i u pravcu x i u pravcu y ose crtali svaku desetu liniju. To je zato što je slika „razvučena" u pravcu y ose na kojoj raspolažemo manjim brojem tačaka. Nešto bolju sliku dobićemo ako u liniji 70 stavimo korak 15, jer je razmera x:y približno 3:2. Poslednja linija ima zadatak da spreči interpretator da nam pokvari sliku porukom OK. Zato izvršenje programa moramo prekinuti pritiskom na <BRK>.

Pramen pravih koje prolaze kroz tačku <0,0> dobićemo ako menjamo koeficijent pravca m u jednačini y=m*x. Doduše, videćemo da program na listingu 3 ne crta jednako kvalitetno sve linije. Znate li zašto?

```
10 CLS
20 FOR M=0 TO 1 STEP .05
30 FOR X=0 TO 511
40 Y=M*X
50 PLOT X,Y
60 NEXT X
70 NEXT M
80 GOTO 80
```

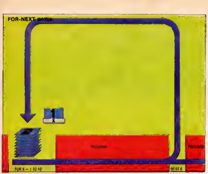
Listing 3

Linija po linija

Videli smo kako možemo nacrtati ma koju pravu liniju. Primetili smo i da svaka ne izgleda jednako dobro kao one paralelne x ili y osi. Da ne bismo eksperimentisali sa raznim formama jednačina prave i vodili računa o greškama zbog približnih izračunavanja koordinata pojedinih tačaka, koristimo naredbu DRAW. Naredba DRAW ima format

DRAW <x>, <y>[, <m>[, <i>]].

Ona crta duž čiji je početak na poziciji grafičkog kursora, a kraj u tački (<x>, <y>) apsolutno ili relativno u zavisnosti od parametra <m>, sa intenzitetom koji se određuje kao u prethodnim naredbama.



Sledeći program predstavlja naulepšan školski primer kako se mogu crati poligoni

```
1   TM X(10) Y(10)
20 [illegible] 'KOLIKO TEMENA (3 DO 10)' N
3  PRINT  UNOSITE REDOM KOORDINATE [illegible]
[illegible] SVIH TEMENA
[illegible]
5  [illegible]
60 NEXT K
7  [illegible]
80 MOVE X(1) Y(1)
90 FOR  =   TO N
100 DRAW X(K) Y(K)
110 NEXT K
1 [illegible] DRAW X(1) Y(1)
140 GOT
```

Listing 4

Skrećemo pažnju da program koji smo naveli nija dovoljno korektan Naime u njemu nija sprečeno unošenja neodgovarajućih ulaznih veličina ali program koji poseduje sva ta svojstva zbog dužine ne bi bio pogodan za unošenja u toku časa.

Krugovi i elipse

Ima više algoritama za crtanja kruga i elipse ali kako su nam ove slike često potrebne u većini jezika sa grafičkim izražajnim mogućnostima za to postoja specijalne naredbe U našem bejziku to je naredba

ELIPSE <x>[, <y>[, <i>]]

Naredba **ELIPSE** crta elipsu sa poluosama dužine (x) i (y) i centrom na poziciji grafičkog kursora Unesite sledeći program

```
10 CLS
20 MOVE 256 128
30 ELIPSE 50 75
```

Ovaj program crta elipsu sa većom poluosom paralelnom y osi Otkucajte sada u direktnom režimu

```
ELIPSE 75 50
```

Ako ste očekivali da će biti nacrtana lata elipse ali sa velikom osom paralelnom x osi pitate se zašto ste dobili krug Setite se da je odnos x y približno 3 2 i da vizuelno praktično dobijamo elipsu sa jednakim poluosama, tj krug Precizno y x=171 256 Ova razmara se podrazumeva ako u naredbi ELIPSE izostavimo vrednost

```
1  [illegible]
   CL
1  MOVE 256 128 0 3
   ELIPSE 180
3  MOVE 17  17  [illegible] 3
40 ELIPSE 40
50 MOVE 340 175 [illegible]
60 ELIPSE
   MOVE 153 58
60 DRAW [illegible]
[illegible]
100 ELIP [illegible]
110 MOVE 340 175 [illegible] 3
120 ELIPSE 20
[illegible]
[illegible]
[illegible]
130 ELIP [illegible]
140 MOVE [illegible]
150 DRAW [illegible]
160 [illegible] NEXT I
16 [illegible]
160 DRAW [illegible] 175
17 GOTO
```

Listing 5

```
1 'stmonj [illegible]
2 CLS
  MOV [illegible]
6 ELIP [illegible]
7 MOVE 30 [illegible]
8 ELIPSE [illegible]
10 MOVE 2 [illegible]
15 ELIPSE [illegible]
16 MOVE [illegible]
17 DRAW  30
18 DRAW  60 [illegible]
20 MOVE 25  90
30 ELIPSE 35 3
40 MOVE 205 15
50 ELIPSE 1
60 MOVE 205 [illegible]
70 ELIPSE [illegible]
80 FOR X=1 T [illegible]
90 NEXT X
100 ELIPS
110 ELIPS
120 ELIPS
130 ELIPSE
140 FOR X=1 [illegible]
150 NEXT X
155 [illegible]
156 [illegible]
157 ELIPS [illegible]
160 GOTO
```

Listing 8

(y) Pogledajte kako izgleda crtanja kruga otkucavši

```
1  BORI [illegible] ZORAN
10 CLS
20 MOVE 25  220 [illegible]
30 ELIPSE 40 20 1
40 MOVE [illegible]
50 DRAW [illegible]
60 MOVE [illegible]
70 ELIPSE 45 50
80 MOVE 2[illegible] 228
90 ELIPSE [illegible] 3 1
100 MOVE 27[illegible]
[illegible]
[illegible]
[illegible]
150 MOVE [illegible]
160 DRAW [illegible]
170 MOVE [illegible]
180 DRAW 286 [illegible]
200 X=1 Y=0
[illegible]
220 DRAW [illegible]
230 MOVE 256 [illegible]
240 ELIP [illegible]
[illegible]
260 ELIPSE 7 5 [illegible]
270 MOVE 25[illegible] 1 0 0 0
280 [illegible]
290 MOV [illegible]
300 DRAW 3[illegible]
310 MOVE 256 [illegible]
320 DRAW 1[illegible]
330 MOVE 256 1[illegible]
340 DRAW 16  23 [illegible]
345 MOVE [illegible]
350 DRAW 360 [illegible]
360 FOR I= [illegible]
370 [illegible]
380 GOTO [illegible]
```

Listing 7

```
1 'SAVA & ENIL
5 CLS
10 MOVE 0 10,0 3
20 FOR I=16 TO 511 STEP 16
21 FOR K=3 TO 0 STEP -3
25 MOVE I 16
30 DRAW -8 -16 1 K
35 MOVE 1 16
40 DRAW 8,-16,1 K
45 MOVE -8 24 [illegible]
49 ELIPSE 8 8,K
50 FILL 0 0 1 K
51 DRAW  16  4 1 K
52 MOVE 32 0 1
53 DRAW -16 4 1 K
55 MOVE 0 12 1
60 ELIPSE 4,4 K
65 NEXT K
70 FOR K=3 TO 0 STEP  3
7  MOVE I+8 16
80 DRAW [illegible] 16 1 K
85 MOVE I+8 16
90 DRAW 2 -16 1 K
95 MOVE 0  4 1
100 ELIPSE 4 8 K
101 FILL 0 [illegible] 1 K
105 MOVE [illegible] 17 1
109 ELIPSE [illegible] 4 K
110 MOVE 0,-12,1
111 DRAW  8  8 1 K
112 MOVE 16 0 1
113 DRAW  [illegible] 8 1 K
115 NEXT K
139 FOR E [illegible] TO 75
140 NEX [illegible]
150 NEXT I
170 REM
```

Listing 6

ELIPSE 100

(Već imamo postavljen grafički kursor na sredinu ekrana naredba ELIPSE ga ne pomera)

Poluose elipse mogu biti brojevi iz intervala [1 2047]. Ako nisu iz tog opsega elipsa se neće nacrtati Ako pak insistirate da se cela elipse vidi na ekranu onda koordinate njenog centra plus/minus odgovarajuće poluose moraju pripadati intervalu [0 511] odnosno [0 255]

Koliko i kako ste naučili da koristite grafičke naredbe možete proveriti ako pokušate da rešite sledeće zadatke

1 Nacrtajte lice koje namiguje

2 Nacrtajte čovečuljka koji radi gimnastiku Tražim li suviše od vas? Pogledajte kako su probleme rešili na drugom času vežbi na računa-

ru moji ubenici Prvi program koji su napisale Ana Simič i Ana Redonjić rešava prvi zadatak (listing 5) Zatim sledi (listing 6) rešenje istog zadatka koje su uredili Vladimir Stanojević i Nikola Tončev Treći program (listing 7) predstavlja rešenje drugog zadataka koja su dali Boris Begenišić i Zoran Balanzada a četvrti (listing 8) je rešenja koje su predložili Emil Božin i Sava Živanović Oni su doduše ubenici trećeg razreda ali kako uče po starom programu ovo im je prvi susrat s računarom

Četvrti susret

Odluka iz čipa

Mogućnost donošenja odluka je najznačajnija prednost računara nad kalkulatorima. Zahvaljujući upravo toj prednosti, računarima se danas prepušta široko polje ljudske delatnosti — od izbora trase kojom će biti probijen novi put do donošenja tako suptilnih odluka kao što su preporuka za životnog saputnika klijentima bračnog savetovališta Kako računar zna da donese neku odluku i koje vrste odluka mogu da donose računari?

Mogućnost donošenja odluka počiva na naredbama grananje koja obezbeđuju upravljanja redosledom izvršavanje Instrukcija programa u zavisnosti od toga da li je ispunjen neki uslov Na primer naredba

```
IF G 18 THEN PRINT MAL [illegible]
```

označava da će poruka MALOLETAN biti štampana jedino ako je vrednost promenljive G koja u ovom kotekstu oznabava godine manja od 18 Ako je vrednost G jednaka ili veća od 18 odnosno ako uslov G<18 nije ispunjen preskočiće se sve što u posmatranom programskom redu sledi Iza službena reči THEN i upravljanje će se preneti na interpretator ili sledeći programski red ukoliko je odgovarajuća naredba sastavni deo programa

IF THEN

IF THEN (ako onda) je jedna od naredbi pranosa upravljanja koja može narušiti uobibajen redosled Izvršavanja instrukcija Podrazumava se naime da se Instrukcija izvršavaju onim redom kojim su fizički smaštene u program Na pobetku se Izvršava prva naredba Kada se njana realizacijá okonča prelazi se na sledeću i tako redom Da bi se ovaj implicitni (koji se podrazumeva) redosled mogao promaniti u bejziku kao i u drugim programskim jezicima imamo mogućnost da instrukcijama dodalimo adrese (leve adrese koje su ovda bpr) na koja zatim pranosimo upravljanje naredbama bezuslovnog prelsska tipa GOTO ili instrukcijama uslovnog prelseka. Ovo je sasvim u skledu sa pravilima koja postoja u jezicima nižeg nivoa u kojima svaki registar koji sadrži naredbu ima edresu a hardvar ume da realizuje naredbe pranosa upravljanja tako što odgovarajuću adresu stavi u brojač naredbi (PC)

GBASIC raspolaže poboljšanom verzijom naredbe IF THEN Njan format je

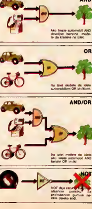
```
IF <uslov> THEN <da grana> ELSE <ne-grana>
```

Ako je (uslov) ispunjen izvršava se (da-grane) programa Ako pak nije ispunjen to jest ako je njegova vrednost netačno onda se izvršava (ne-grana) programa (da–grana) i (ne–grana) mogu biti niz instrukcija ili samo broj programskog reda Po izvršenju naredbi iz (da–grane) ili (ne–grane) upravljanje se prenosi na sledači programski red Ako je pak umesto naredbi u nekoj od grana samo broj programskog reda upravljanje se prenosi na liniju sa tim brojem Prednost naredbe IF THEN ELSE u odnosu na običnu IF THEN je što pragledno prikazuje koja akcija preduzimamo u slučaju da je uslov zadovoljen a koja ako nije Njano dajstvo ilustrujemo programom koji računa prosečnu ocenu iz nekog predmeta

```
[illegible]
[illegible]NT AKO [illegible] VIŠE POL[illegible]
[illegible] PROSEČNA OCENA [illegible]
[illegible] SUMA=0 BROJ=0
[illegible] INPUT OCENA
60 IF OCENA -1 THEN PRINT [illegible]
/BROJ STOP ELSE SUMA SUMA [illegible]ENA
BROJ BROJ 1
70 [illegible]
```

U liniji 30 znakovnoj promenljivoj P$ dodaljujemo kao vrednost poruku koju želimo da štampamo uz prosečnu ocenu Ovom rešenju smo pribegli da bi složena linija 60 bila čitljivija Kada program naiđe na liniju 60 u zavisnosti od broja koji smo uneli štampa prosečnu ocenu i prekida rad ili sabira tu ocenu sa prethodno unetim i uvećava broj unetih ocene za 1

Startujte program i unesite podatka 5,2 –1 jedan po jedan posla postavljenog znaka pitanja Poslednji podatak aktivirao je (da– granu) u liniji 60 pa dobijate poruku




```
PROSEČNA OCENA JE 3 5
Break in 60
```

Obaranje programa

Startujte ponovo program i odmah unesite vrednost –1 Dobijate poruku

```
PROSEČNA OCENA JE Division by zero
1 70141E+38
```

Problem je u tome što nismo predvidali mogućnost da korisnik ne unese ni jednu ocenu ili pogreši Program je izvršio deobu 0/0 i prijavio nam tu neregularnu situaciju Štampao je i najveći broj koji može da zabeleži 1 70141E+38 jer što je delilac manji to je imenilac veći Za računar su svi brojevi manji od 10I–38 mašinska nula a svi veći od malopre odštampanog beskonačno veliki jer ne može da ih zabeleži na način na koji on pamti brojeve Zato ćemo pokušati da grešku ispravimo na sledeći način Liniju 60 izmenićemo u

```
60 IF OCENA= 1 THEN 80 ELSE SUMA=
=SUMA+OCENA BROJ=BROJ+1
— Dodaćemo i linije 80 90 i 100
80 IF B=0 THEN PRINT MORATE UNETI
BAR JEDNU OCENU GOTO 50
90 PRINT P$ SUMA/BROJ
10[illegible]
```

Tako izmenjanom programu na samom početku unesite –1 On neće pasti jer smo se u liniji 80 obezbedili od deljenja nulom Uvek kada pišete programe morate voditi računa i o ovakvim stvarima

Linija 90 ilustruja da u naredbi IF možemo izostaviti službenu reč ELSE i (ne–granu) Takođe se može službena reč THEN zameniti sa GOTO ako u (da–grani) navodimo samo broj programskog reda

Složeni uslovi

Ako red programa zavisi od više uslova problem se može rešiti pisanjem više uslovnih naredbi ili kombinovanjem uslova u jednoj IF THEN naredbi pomoću logičkih operatora U našem programu recimo problem je nastupao kada je OCENA=–1 i BROJ=0 što smo mi utvrđivali pomoću dve IF THEN naredbe Moguće je bilo smestiti obe IF THEN naredbe u isti programski red ali ovakvo rešenja je obično nepregledno i bolje je da ga izbegavate Ostaje jedino dilema da li pisati više IFove ili jedan sa složenim uslovom Programeri koji ne znaju mašinski jezik obično smatraju da će se brže izvršiti jedna IF THEN naredba sa složenim uslovom nego više njih sa prostim To nije tačno Naime u mašinskom jeziku preko kojeg se interpretiraju sve instrukcije bejzika postoje samo prosta naredbe prenosa upravljanja sa jednom adresom Shodno tome provera složenog uslova mora se svoditi na odgovarajući broj jediničnih mašinskih instrukcija granenja S druge strane ako umesto složena provera u jednoj IF THEN naredbi zapišemo više jednostavnih IF naredbi odgovarajućim redosledom provera možemo značajno smanjiti vrema izvršavanja programa Na primer ako imamo zapise tri tipa sa kodovima 1 2 i 3 i 90% ih pripada tipu 1 9% tipu 2 i 1% tipu 3 onda bi najbolje bilo postavljati pitanja ovim redom

```
[illegible]
[illegible]
[illegible]
```

Sa ovakvim redosledom pitanja u 90% slučajeva ispitivaće se samo jedan uslov u 9% dva uslova i samo u 1% tri uslova Prosečno će se po kodu proveravati

$$9\cdot1+09\cdot2+01\cdot3=1\ 1$$

uslov a u slučaju ako bismo postavljali obrnut red pitanja po kodu bi se proveravalo

$$01\cdot1+09\cdot2+9\cdot3=2\ 89$$

uslova Konkretno ako bi se klijent obratio elektronskoj provodadžiki sa zahtevom da mu odabere nevestu koja je mlada lepa pametna, nevina i ima 1 000 000 dolara miraza najbrži odgovor bi dao program koji bi poslednju želju postavio kao prvo pitanje

Ako bez obzira na brzinu izvršavanja programa više volite da imate pregledno zapisane uslove koristićete logičke operatore Upoznajmo zato mogućnosti reda našeg interpretatora sa logičkim izrazima

Konstante i izrazi

Za razliku od drugih viših programskih jezika bejzik nama ugrađenu mogućnost za rad sa konstantama tipa tačno' i netačno Medutim on je sa uspehom simulira Po njemu, brojna vrednost 0 predstavlja logičku konstantu „netačno a svaka druga logičku konstantu , tačno Uverite se u to sledećim malim programom

```
10 INPUT A
20 IF A THEN PRINT  TAČNO  ELSE PRINT NETAČNO
30 GOTO 10
```

Za sve vrednosti promenljive sem za 0 program će štampati TAČNO

Medutim samo registrovanja logičke konstante tačno razlikuje se od računara do računara Obično se beleži kao broj 1 ili −1 Proverite kako vaš računar beleži logičke konstante otkucevši najpre

PRINT A=A

što će vam pokazati kako se pamti tečno , a zatim

PRINT A<>A

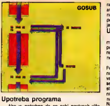
što će vam pokazati kako se pamti , netačno Kod , tima , tačno se pamti kao broj −1 (što se u registru beleži kao 11111111) a netačno kao 0 (u registru kao 00000000)

Promenljiva koja može dobiti samo vrednost 0 1 −1 za nas je logička promenljiva Logičke konstante i logičke promenljive su logički izrazi Ako su P i q logički izrazi onda su i

```
NOT P i NOT q
P AND q
P OR q
P EOR q
P IMP q
P EQV q
```

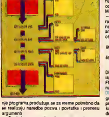
takođe logički izrazi čija se vrednost izračunava po odgovarajućim tablicama istinitosti Operator NOT je oznaka za negaciju AND za konjunkciju OR za disjunkciju EOR za ekskluzivnu disjunkciju IMP za implikaciju i EQV za ekvivalenciju Verovatno znate da izračunate vrednosti ovih logičkih izraza ali to vam ne smeta da unesete i koristite program LOGIČKI OPERATORI sa slike 9 koji štampa odgovarajuće tablice istinitosti pogotovu što ćemo ga iskoristiti kao primer koji ilustruja korišćenje potprograma

```
10 DEF FNA(P,Q)=P OR Q  DEF FNB(P,Q)=P AND Q
20  DEF FNC(P,Q)=(P AND NOT Q) OR (NOT P AND Q)
30 DEF FND(P,Q)=P IMP Q  DEF FNE(P,Q)= P EQV Q
40 FOR I=1 TO 5: READ A$(I)
50 ON 1 GOTO 60,70,80,90,100
60 DEF FNO(P,Q)=FNA(P,Q): GOTO 110
70 DEF FNO(P,Q)=FNB(P,Q): GOTO 110
80 DEF FNO(P,Q)= FNC(P,Q): GOTO 110
90 DEF FNO(P Q)=FND(P,Q): GOTO 110
100 DEF FNO(P,Q)=FNE(P,Q)
110 GOSUB 160
120 PRINT
130 O$=INKEY$:IF O$="" THEN 130
140 NEXT I
150 STOP
160 PRINT "FUNKCIJA ";A$(I)
170 FOR P=0 TO -1 STEP -1
180 FOR Q=0 TO -1 STEP -1
190 PRINT P,Q;FNO(P Q)
200 NEXT Q
210 NEXT P
220 RETURN
230 DATA OR,AND,EOR,IMP,EQV
```



Upotreba programa

Ako je potrebno da se neki postupak više puta ponovi u programu kao što je trebalo da se kod nas pet puta ponovi dvostruki ciklus po p i q onda je zgodno da se naredbe koje ga opisuju izdvoje u posebnu celinu — potprogram Potprogram bi se pozivao iz svake tačke programa u kojoj je neophodno primeniti taj postupak i po izvršenju vraćao bi upravljanje glavnom programu na naredbu koja sledi neposredno iza poziva Ovakva organizacija programa omogućava njegov kreći zapis Ako je opis postupka zahtevao n programskih redova i ponavljao se za razne argumente m puta u programu onda umesto n*m programskih redova bez korišćenja potprograma pišemo samo n redova (+m poziva i bar m povrataka u program) Zahvaljujući toma smanjuja se mogućnost greške pri pisanju programa olakšano je testiranja njegova ispravnosti i štedi se memorijski prostor Ali sve ima svoju cenu — gubi se na brzini izvršavanja programa Izvršavanje programa produžuje se za vreme potrebno da se realizuju naredbe poziva i povratka i prenesu argumenti



Dosledno sprovedena koncepcija potprograma podrazumeva da su oni potpuno izdvojene celina nezavisne od glavnog programa. To znači da se mogu koristiti iz raznih programa i da imaju sopstvena lokalne promenljive koje nemaju nikakve veze sa istoimenim promenljivim glavnih programa Ova činjenica omogućava timski rad u pisanju aplikacija jer je dovoljno da jedan programer da globalni opis rešenja a razradu detalja može prepustiti svojim kolegama koji će to učiniti u zasebnim potprogramima Glavni programer projektuje samo veza između ovih celina i naznačava koje su promenljive globalne to jest zajedničke za program i pojedine potprograme

Bejzik o kome govorimo nema ovu mogućnost što je njegova velika mana Ono čime raspolažemo jesu mogućnost definisanja korisničkih funkcija pomoću takozvane funkcijske naredbe i korišćenje internih potprograma — potprogramskih segmenata koji mogu da stoje jedino u okviru glavnog programa.

Uvedene funkcije

Računari obično imaju ugrađene potprograme za računanje elementarnih funkcija koji se pozivaju navođenjem imena funkcija sa konkretnim argumentima za koje treba izračunati vrednost Naredba

PRINT SIN(0)

Poziva ugrađeni potprogram za računanje vrednosti funkcije sinus • Argument 0 je ulazna veličina u potprogram stvarni argument za koji se računa vrednost funkcije po ugrađenom postupku opisanom za fiktivne argumente Izračunata vrednost se zatim vraća u program (ovde u naredbu u neposrednom režimu rada) i štampe Nemaju svi računari ni sve verzije bejzika isti broj ugrađenih funkcija niti je računanje njihovih vrednosti istog kvaliteta. U GBASICu imamo ukupno 44 ugrađene funkcije pri čemu neke od numeričkih daju rezultat u dvostrukoj preciznosti Medutim detaljna analiza njihovog rada pokazuje da ni one ne daju potpuno korektne rezultate, pa je za one koji žele tačnija računanja jedino rešenja da sami uvedu svoje postupke za njihova određivanja Isto tako, ni svi logički operatori u našem bejziku ne rade ono što bi trebalo U priručniku piše da se ekskluzivna disjunkcija označava sa XOR Proverite u direktnom režimu šta dobijete ako tražite

? 0 XOR 0

Trebalo bi da dobijete 0 a računar štampa 0 0 0 znači operator XOR tumači kao promenljivu sa imenom XOR kojoj prethodno nije dodeljena vrednost. Ako pak tražite

^ 0 EOR 0

Kako se ekskluzivna disjunkcija označava kod nekih drugih računara dobićete samo jednu nulu To znači da je računar sada shvatio' da od njega tražimo da sračuna vrednost izraza Medutim ako proverite kako se računa vrednost −1 EOR −1 videćete da ipak nemamo na raspolaganju ekskluzivnu disjunkciju već je računar uzimao u obzir druga dva slova a E je smatrao sastavnim delom broja Uverite se u to otkucavši,

? −1 E

što će kao rezultat dati −1 a zatim

I=−1 J=−1 ?I EOR J

što će kao i pri korišćenju XOR dati poruku

−1 0 −1

Zato u našem programu ne kažemo samo DEF ENO<P.Q>= P EOR O, već prethodno ispravno definišemo ovu funkciju pod imenom FNC Korisnička funkcije definišu se posebnom naredbom oblika

DEF FN Ime [(lista)]=<izraz>

gde se <ime> formira po istim pravilima kao ime promenljive a <lista> je spisak promenljivih koja će biti upotrebljene u <izrazu> kojim se definiše funkcija U <listi> su fiktivna promenljiva koja samo naznačavaju kakav tip promenljive će biti upotrebljen Navođenjem imena uvedene funkcije sa konkretnim argumentima biće izračunata vrednost <izraza> pri čemu se odgovarajući fiktivni argumenti zamenjuju stvarnima. Naredbe DEF FN mora biti izvršena pre no što se funkcija pozove u suprotnom se dobija poruka o grešci Napominjemo da naš bejzik ima dosta dobre mogućnosti za uvođenje funkcija, jer se mogu definisati i numeričke i znakovne funkcije više promenljivih a kao što iz primera vidite uspeli smo da uvedemo i logičke funkcije Kod komodora 64 na primer moguće je uvođenje samo numeričkih funkcija jedne promenljive No te mogućnosti bi mogle biti i bolje kada bi se kao kod BBC bejzika funkcije mogla definisati rekurzivno

Snimite prethodni program pa otkucajte NEW i proverite šta radi sledeći program

```
10 DEFFNA(A B C, D)=A•B•C
20 PRINT „UNESITE DVA BROJA
30 INPUT A, B
40 PRINT FNA (A B)
```

Posle startovanja programa videćete da on — ne radi Promenite liniju 40 na sledeći način

```
40 PRINT FNA (A B 0 0 0)
```

I program će raditi Upamtite da se prilikom svakog poziva funkcije moraju navesti stvarni umesto svakog fiktivnog argumenta

Potprogramski segmenti

Mnogo veće mogućnosti na svim računarima pružaju potprogramski segmenti posebni delovi programa na koje se upravljanje prenosi naredbom

GOSUB <broj reda>

Postoji značajna razlika između naredbi GOTO i GOSUB. Realizacija GOTO naredbe podrazumeva da se na mesto gde se čuva adresa sledeće naredbe stavi adresa na koju ukazuje GOTO GOSUB pak uz sve ono što proizvodi GOTO još i čuva adresu naredbe koja je bila zapisana neposredno iza nje (upisuje je u stek) Ovo omogućava da se po izvršenju potprogramskog segmenta osveži sećanje na mesto odakle je pozvan potprogram i vrati mu se upravljanje Taj povratak u program ostvaruje se naredbom RETURN (to nije tipka RET) koja vraća upravljanje na poslednju adresu ipisanu u stek.

Dok korišćenje GOSUB naredbi u programu obično govori o njegovoj dobroj strukturi dotle mnoštvo instrukcije bezuslovnog prelaska svedoči o njegovoj lošoj razredi Program sa mnoštvom naredbi GOTO teško se testira i razume Osnovnu teškoću predstavlja praćenje strukture takvih programa jer nas pri analizi programa svaka GOTO naredba stavlja na muku može da prenese upravljanje u ma koji deo programa a odakle — to možemo videti jedino ako imamo ceo listig pred očima Mada ima jazika u kojima se može programirati bez GOTO instrukcija,u našoj verziji bejzika to nije moguće

Višestruko grananje

U programima se često traži da u zavisnosti od vrednosti neke promenljive upravljanje možemo da prenesemo na više od dve grane programa Tako recimo u zavisnosti od toga da li je vrednost nekog brojnog izraza negativna jednaka null ili pozitivna, možemo preduzeti tri različite akcije U našem programu, u zavisnosti od vrednosti promenljive I preduzimali smo pet različitih akcija Ako je I bilo jedan, pozivao se potprogram u liniji 160 da sračuna vrednost prve funkcije u slučaju da je I=2 tražilo se računanje i štampanje vrednosti duge uvedene funkcije i tako redom Ovakvo grananje smo razume se mogli da ostvarimo pomoću više IF THEN naredbi ali smo pribegli elegantnijem rešenju — koristili smo On GOTO

Naredba ON GOTO ima format

ON <brojni izraz> GOTO <n1> <n2>, <nK>

Bejzik interpretator nailaskom na ovu naredbu najpre izračunava celobrojnu vrednost brojnog izraza između ON i GOTO Ako je ta vrednost 1, prenosi upravljanje na programski red <n1> ako je 2 na programski red <n2> ako je K na K-ti po redu broj programskog reda iz liste koja sledi iza službene reči GOTO Ako odgovarajući brojni izraz ima vrednost manju od 0 ili veću od 255 program se prekida uz izveštaj Illegal Function Call Ako je vrednost brojnog izraza 0 ili veća od broja linija navedanih u listi iza GOTO upravljanje se prenosi na programski red koji sledi neposredno iza ON GOTO

Program „Logički operatori"

U programu logički operatori na samom početku uvodimo 5 logičkih funkcija naredbom DEF FN Za konjunkciju disjunkciju implikaciju i ekvivalenciju za koje logički operatori AND OR, IMP i EQV korektno rade uveli smo samo imena FNB FNR FND i FNE Za ekskluzivnu disjunkciju koju smo u programu nazvali FNC morali smo navesti kako se računa preko NOT, AND i OR Ove tri funkcije su značajne jer se pomoću njih može predstaviti svaka logička funkcija Ako znate tablicu vrednosti neke logičke funkcije možete zapisati i njen analitički oblik u tako zvanoj savršenoj disjunktivnoj normalnoj formi u kojoj se pojavljuju samo operatori NOT AND i OR Tako smo i mi došli do zapisa funkcije EOR za koju znamo da je „tačna" kada su argumenti različiti a „netačna" kada su argumenti jednaki

U ciklusu od 40 do 140 linija pet puta se poziva potprogramski segment koji počinje u liniji 160 Potprogramskim segmentom je rešeno štampanje tablice vrednosti funkcija sa imenom A$ definisana formulom FNO <P Q> Odgovarajuće ime čita se iz DATA linije (detaljnije o upotrebi naredbi READ DATA biće reči kasnije) a pravilno FNO se u zavisnosti od vrednosti ciklusne promenljive I pomoću naredbe višestrukog grananja, definiše na jedan od pet načina datih u linijama 60—100 Na kraju ciklusa po povratku iz potprograma naredbe PRINT štampanjem praznog reda povećava preglednost ispisa a linije 130 ne dozvoljava da se nastavi red programa sve dok korisnik pritiskom na neku tipku ne stavi do znanja da može da prati dalji ispis

Zadaci za vežbu

1 Pokušajte da napravite odgovarajuće programe koji bi štampali tablice vrednosti funkcija NI i NILI značajnih po tome što svaka od njih predstavlja pun sistem funkcija, to jest preko njih se može izraziti bilo koja druga logička funkcija. (NI je negacija funkcije AND i u engleskoj literaturi se zova NAND NILI ili NOR predstavlja negaciju disjunkcije)

2 Ako znate šta je tautologija pokušajte da sastavite program koji bi proveravao da li dva zadata logička izraza povezana znakom jednakosti predstavljaju tautologiju

3 Iskoristite svoje poznavanje grafičkih naredbi i logičkih funkcija da sastavite program koji vizuelno prikazuje značenje svake od njih

Peti susret

S bejzikom na ti

Govoreći o zapisu složenih uslova pomoću naredbi IF . . . THEN upoznali smo se i sa logičkim operatorima. Njihova glavna upotreba, međutim, vezuje se za neposredan rad sa sadržajima memorijskih registara. Ove uobičajene postupke pri programiranju u simboličkom jeziku iz bejzika možemo realizovati pomoću naredbe POKE i funkcije PEEK koju smo već upoznali.

Podsetimo se da smo upoznali ukupno šest logičkih operatora NOT, AND OR EOR IMP i EQV Šta ćemo dobiti ako tražimo

? 1 AND 0

Razume se 0 a kad tražimo

? 1 OR 0

dobijemo 1

No šta se dešava kad zatražimo

? 1 EQV 0

Umesto 0 dobijamo vrednost −2. Šta je po sredi?

Računar sva logička operacije izvršava nad svakim bitom registra pojedinačno Vodite računa da je u našoj implemetaciji bejzika tačno =−1 a ne 1 U poslednjem primeru računar primenjuje logički operator EQV nad bajtovima sa sadržajima 00000001 i 00000000 Kao rezultat se dobija 11111110 što predstavlja potpuni komplement broja −2 No da bi sve bilo jasnije moramo prethodno poznavati računarov maternji jazik — binarni brojni sistem Računari koriste binarni sistem računanja zbog toga što u odnosu na druge brojne sisteme raspolaže nizom prednosti Njegove cifre lako se predstavljaju pomoću električnih signala a za njihovo memorisanje potrebni su jednostavni dvopozicioni elementi U njemu se lako i brzo izvršavaju aritmetičke i logičke operacije Uz to korišćenje binarnog brojnog sistema dozvoljava da se količina elektronskog materijala svede na minimum

Pravođenje brojava

Računari ulazne veličine moraju preveeti iz dekadnog brojnog sistema koji mi koristimo u binarni u kome su oni u stanju da računaju Takođe dobijene binarne rezultate treba da vrate u dekadni sistem kako bi ih mogli upotrebljavati i korisnici koji ne poznaju finese mašinskog programiranja Želimo li da pratimo kako računar radi, moramo umeti da pravodimo brojeva zapisane u jednom pozicionom sistemu u pozicioni sistem druge osnove



U principu broj N zapisan u brojnom sistemu osnove A može se prevesti u brojni sistem osnove B na dva načina vršenjem aritmetičkih operacija u polaznom ili vršenjem aritmetičkih operacija u rezultujućem brojnom sistemu Prvi način biramo kada prevodimo iz dekadnog u neki drugi brojni sistem Drugom načinu pribegavamo kada treba broj prevoditi u dekadni sistem jer tada radimo po aritmetičkim pravilima na koja smo navikli Ilustrovaćemo ovo primerom Neka treba broj 344 201 iz osnove 5 prevesti u osnovu 10 Imamo

$$(344\ 201)_5 = 3\cdot5^2+4\cdot5^1+4\cdot5^0+2\cdot5^{-1}+0\cdot5^{-2}+1\cdot5^{-3} = 75+20+4+0\ 4+0\ 008 = (99\ 408)_{10}$$

Dakle, broj N zapisan u sistemu osnove A zapisujemo u razvijenom obliku pri čemu sve cifre osnovu i odgovarajuće stepene zapisujemo onako kako se pišu u dolaznom brojnom sistemu i izvršimo naznačene operacije po aritmetičkim pravilima rezultujućeg brojnog sistema

Ako pak treba da prevedemo broj iz dekadnog u neki drugi brojni sistem, više bi nam odgovaralo da radimo u polaznoj osnovi U ovom slučaju razlikuje se postupak za prevođenje celih brojeva od prevođenja razlomaka Ceo broj prevodi se tako što se izvrši celobrojna deoba osnovom rezultujućeg sistema, upamti se ostatak a postupak sa ponovi sa količnikom Postupak je završen kada se dobije količnik 0 a zapis broja u novom sistemu predstavlja niz ostataka zapisanih u obrnutom redosledu od redosleda dobijanja

```
699 ' ***IZDEK*** PREVODJENJE DEKADNOG B
ROJA N U OSNOVU B
700 IF INT(N)<>N OR N<0 THEN PRINT "POTP
ROGRAM IZDEK RADI SAMO ZA POZITIVNE CELE
 BROJEVE": RETURN
710 IF INT(B)<>B OR B<2 OR B>36 THEN PRI
NT "POTPROGRAM IZDEK MOZE PREVESTI SAMO
U SISTEM OSNOVE 2 DO 36". RETURN
720 N$=""
730 M=N\B:R=N MOD B: N=M
740 IF R<10 THEN N$=CHR$(48+R)+N$ ELSE N
$=CHR$(55+R)+N$
750 IF N<>0 THEN 730
760 RETURN
```

Listing 10

Potprogram IZDEK prikazan na sl 10 radi na ovaj način Za zadatu osnovu B on proizvodi zapis N$ zadatog dekadnog broja N Program počinje od linije 700 i najpre proverava da li je pozvan sa korektnim ulaznim veličinama to jest zahteva da za prevođenje dobija ceo pozitivan dekadni broj i celobrojnu osnovu između 2 i 36

Recimo pri prevođenju dekadnog broja 93 u binarni sistem imali bismo sledeće vrednosti promenljivih

	M	R	N	N$
Početno	0	0	93	""
1 Prolaz	46	1	46	"1"
2 Prolaz	23	0	23	"01"
3 Prolaz	11	1	11	"101"
4 Prolaz	5	1	5	"1101"
5 Prolaz	2	1	2	"11101"
6 Prolaz	1	0	1	"011101"
7 Prolaz	0	1	0	"1011101"

Dakle $(93)_{10}=(1011101)_2$

Potprogram IZDEK u liniji 730 koristi dva celobrojna operatora To su celobrojno deljenje koje se označava obrnutom kosom crtom (\) i ostatak pri celobrojnom deljenju koji se označava sa MOD Kako ovi operatori mogu da se primenjuju samo nad brojevima iz intervala [−32768 32767] to bi potprogram IZDEK imao širu primenu ako bi u liniji 730 stajalo

730 M=INT(N/B) R=N/B−M

Pribegli smo ovom lošijem rešenju jer smo želeli da naglasimo da „tim" može da radi sa celobrojnim operatorima Po prioritetu je celobrojno deljenje odmah iza množenja i običnog deljenja a MOD je neposredno iza njega (Vidite prilog Prioritet operatora) Memorišite ovaj program sa SAVE "IZDEK",A

Za razliku od celih brojeva razlomljeni ne mogu uvek tačno da se prevedu Stoga moramo zadavati tačnost $B^{(-A)}$ Kada odredimo A mesta iza pozicione tačke prestajemo da prevodimo i smatramo da smo obavili prevođenje sa zadatom tačnošću To prevođenje se vrši tako što se polazni broj (0<R<1) množi osnovom B ceo deo se pamti a postupak nastavlja se dobijenim razlomljenim delom rezultata Cifre broja u novom sistemu predstavljaju upamćene celobrojne vrednosti zapisane u redosledu dobijanja

```
799 ' ***BINRAZ*** PREVODJENJE DEKADNIH
U BINARNE RAZLOMKE
800 IF INT(R)=R OR R<0 THEN PRINT "POTPR
OGRAM BINRAZ RADI SAMO ZA PRAVE POZITVNE
 RAZLOMKE": RETURN
810 A=INT(A): IF A<0 THEN PRINT "PONOVIT
E NA KOLIKO CIFARA TREBA IZRACUNATI REZU
LTAT": RETURN
820 R$=""
830 FOR T=1 TO A
840 R=R*2
850 R$=R$+MID$(STR$(INT(R)),2,1)
860 R=R-INT(R)
870 IF R=0 THEN RETURN
880 NEXT T
890 RETURN
```

Listing 11

Potprogram BINRAZ (slika 11) radi na ovaj način Za zadat broj proizvodi odgovarajući binarni zapis U ovom slučaju predvideli smo da postupak prekine ako se dobije tačan prevod ili posle određivanja A cifara iza binarne tačke

Potprogram BINRAZ počinje u liniji 800 i kao i IZDEK prihvata samo regularne ulazne veličine Kako radi ilustrovaćemo na primeru prevođenja broja 0 157 sa tačnošću A=5 binarnih cifara Imamo sledeće vrednosti promenljivih

	T	R	R$	R
Početno	0	157	""	157
1 Prolaz	1	314	"0"	314
2 Prolaz	2	628	"00"	628
3 Prolaz	3	1 256	"001"	256
4 Prolaz	4	512	"0010"	512
5 Prolaz	5	1 024	"00101"	024

Memorišite i ovaj program sa SAVE "BINRAZ",A

Očistite sada memoriju i pripremite se za upoznavanje novih mogućnosti bejzik interpretatora

Nove olakšice

Olakšice pri pisanju programa pružaju i komande AUTO RENUM i MERGE Prva automatski ispisuje brojeve programskih redova druga može da prenumeriše postojeći program a treća povezuje program iz memorije sa programima koji se čuvaju na disku u aski formatu

Komanda AUTO ima format AUTO [<broj reda>[,<uvećanje>]] Njome se postiže automatska numeracija programskih redova počev od <broj reda> sa <uvećanjem> Recimo AUTO 50,5 će generisati linije sa brojevima 50 55 60 sve dok se CTRL/C ne prekinemo njeno dejstvo

Otkucajte AUTO 10 10 i ispišite sledeći program (slika 12)

```
5 CLS
10 INPUT "UNESI CEO DEKADNI BROJ";X
20 PRINT "UNESI OSNOVU B.S U KOJI ZELIS
 DA PREVEDES BROJ",
30 INPUT B
40 CLS PRINT
50 N=X GOSUB 700
60 PRINT "DEKADNI BROJ";X;" U OSNOVI",B,
"IZNOSI "N$
70 TEXT "PRITISNITE BILO KOJU TIPKU ZA
NASTAVAK",150,120
80 P$=INKEY$ IF P$="" THEN 80
90 GOSUB 900
100 END
```

Slika 12

Zatim u direktnom režimu otkucajte MERGE "IZDEK", a zatim MERGE "KRAJ"

Ako sada zatražite LIST videćete da u memoriji imate sledeći program (slika 13)

```
5 CLS
10 INPUT "UNESI CEO DEKADNI BROJ",X
20 PRINT "UNESI OSNOVU B.S, U KOJI ZELIS
 DA PREVEDES BROJ",
30 INPUT B
40 CLS:PRINT
50 N=X GOSUB 700
60 PRINT "DEKADNI BROJ";X;" U OSNOVI";B,
"IZNOSI "N$
70 TEXT "PRITISNITE BILO KOJU TIPKU ZA
NASTAVAK",150,120
80 P$=INKEY$: IF P$="" THEN 80
90 GOSUB 900
100 END
699 ' ***IZDEK*** PREVODJENJE DEKADNOG B
ROJA N U OSNOVU B
700 IF INT(N)<>N OR N<0 THEN PRINT "POTP
ROGRAM IZDEK RADI SAMO ZA POZITIVNE CELE
 BROJEVE": RETURN
710 IF INT(B)<>B OR B<2 OR B>36 THEN PRI
NT "POTPROGRAM IZDEK MOZE PREVESTI SAMO
U SISTEM OSNOVE 2 DO 36": RETURN
720 N$=""
730 M=N\B:R=N MOD B: N=M
740 IF R<10 THEN N$=CHR$(48+R)+N$ ELSE N
$=CHR$(55+R)+N$
750 IF N<>0 THEN 730
760 RETURN
900 CLS: TEXT "ZELITE LI DA PREKINETE S
RADOM (D/N)",150,120,0,3
910 O$= INKEY$
920 IF O$<>"D" AND O$<>"N" GOTO 910
930 IF O$="N" THEN RUN
936 FOR I=1 TO 5
940 CLS TEXT "ONDA, DOVIDJENJA ",200,100
,0,1
950 TEXT "ONDA, DOVIDJENJA ",199,101,0,3
960 FOR J=1 TO 500: NEXT J: TEXT "ONDA,
DOVIDJENJA",200,100,0,0: TEXT "ONDA, DOV
IDJENJA ",199,101,0,0
970 NEXT I
```

Slika 13

Komanda koja povezuje programe sa diska sa programom u memoriji ima format MERGE (ime programa) Program koji se poziva mora na disku biti upamćen u aski formatu a ako nije ispisuje se poruka Bad file mode Budite oprezni jer ukoliko u programu na disku i programu u memoriji ima linija sa istim brojem programske linije u memoriji biće zamenjene linijama sa diska.

Proverite šta će se desiti kad otkucate RENUM 10 10

Iskoristimo potprograme koje imamo za kreiranje još jednog složenog programa koji vam može pomoći u uvežbavanju prevođenja dekadnih u binarne brojeva Glavni program treba da generiše zadatke postavlja pitanja i pomoću navedenih potprograma proverava da li su vaši odgovori korektni Očistite memoriju i unesite sledeći program (slika 14)

```
10 X=INT(RND*100) Y=RND
20 N=X B=2 GOSUB 700
30 R=Y A=6 GOSUB 800
40 CLS PRINT
50 PRINT"KOLIKO U BINARNOM SISTEMU IZRAC
UNATO NA 6 DECIMALA IZNOSI DEKADNI BROJ"
60 PRINT TAB(40),X+Y
70 INPUT O$
80  IF O$=N$+R$ THEN PRINT "TACNO" ELSE
PRINT "NETACNO, ISPRAVNO JE ",N$+R$
90 TEXT "PRITISNI BILO KOJU TIPKU ZA NAS
TAVAK",150,120,0,3
100 P$=INKEY$: IF P$="" THEN 100
110 GOTO 10
```

Slika 14

Povežite ovaj program sa potprogramima IZDEK i BINRAZ pomoću MERGE Iste potprogra-

me možete iskoristiti i za suprotan zadatak za uvežbavanje prevođenja iz binarnog u dekadni brojni sistem. Vodite računa da provere da li ste dali korektan odgovor pri korišćenju ovih programa neće moći da se realizuje pomoću znaka jednakosti. Kriterijum za utvrđivanje da li je odgovor dobar mora biti provera uslova da li je postignuta tačnost na određen broj decimala. Kada budemo detaljnije govorili o znakovnim funkcijama daćemo rešenje prevođenja iz binarnog u dekadni brojni sistem koje je apsolutno tačno.

```
10 X=INT(RND*100) Y=RND
20 N=X B=2 GOSUB 130
30 R=Y A=6 GOSUB 210
40 CLS PRINT
50 PRINT"KOLIKO U BINARNOM SISTEMU IZRAC
UNATO NA 6 DECIMALA IZNOSI DEKADNI BROJ"
60 PRINT TAB(40);X+Y
70 INPUT O$
80  IF O$=N$+R$ THEN PRINT "TACNO" ELSE
PRINT "NETACNO, ISPRAVNO JE ",N$+R$
90 TEXT "PRITISNI BILO KOJU TIPKU ZA NAS
TAVAK",150,120,0,3
100 P$=INKEY$: IF P$="" THEN 100
110 GOTO 10
120 ' ***IZDEK*** PREVODJENJE DEKADNOG B
ROJA N U OSNOVU B
130 IF INT(N)<>N OR N<0 THEN PRINT "POTP
ROGRAM IZDEK RADI SAMO ZA POZITIVNE CELE
 BROJEVE": RETURN
140 IF INT(B)<>B OR B<2 OR B>36 THEN PRI
NT "POTPROGRAM IZDEK MOZE PREVESTI SAMO
U SISTEM OSNOVE 2 DO 36": RETURN
150 N$=""
160 M=N\B R=N MOD B N=M
170 IF R<10 THEN N$=CHR$(48+R)+N$ ELSE N
$=CHR$(55+R)+N$
180 IF N<>0 THEN 160
190 RETURN
200 ' ***BINRAZ*** PREVODJENJE DEKADNIH
U BINARNE RAZLOMKE
210 IF INT(R)=R OR R<0 THEN PRINT "POTPR
OGRAM BINRAZ RADI SAMO ZA PRAVE POZITVNE
 RAZLOMKE": RETURN
220 A=INT(A): IF A<0 THEN PRINT "PONOVIT
E NA KOLIKO CIFARA TREBA IZRACUNATI REZU
LTAT" RETURN
230 R$=" "
240 FOR T=1 TO A
250 R=R*2
260 R$=R$+MID$(STR$(INT(R)),2,1)
270 R=R-INT(R)
280 IF R=0 THEN RETURN
290 NEXT T
300 RETURN
```

Primetili ste da ovom prilikom u program nismo uključili potprogram KRAJ. Proverite šta se dešava ako u liniji 110 umesto GOTO 10 stavite GOSUB 900. Zbog novog RUN u okviru potprograma funkcije RND će stalno generisati isti slučajni broj pa će se stalno postavljati isti zadatak. Razume se postoji rešenje ovog problema o njemu neki drugi put.

Razmotrimo sada detaljnije dejstvo komandi AUTO i RENUM jer njihova primena u nekim drugim bejzicima u specijalnim slučajevima pravi velike probleme. Tako recimo u Sejmons bejziku komanda AUTO ne poznaje granice numerisanja prihvata korak numeracije 0 i dozvoljava da ako u memoriji prethodno već postoji neki program upisujete nove linije sa istim bpr. Ukoliko AUTO kod našeg bejzika generiše broj linije koja već postoji iza broja linije se pojavljuje zvezdica kao upozorenje. Ako tada pritisnete RET linije neće biti promenjena a AUTO će generisati sledeći broj.

Ukoliko se parametri izostave podrazumeva se da su oba 10 to jest biće generisane linije 10 20 30 ... Naša komanda AUTO poznaje granicu numerisanja (65529) i ne dozvoljava da se ta granica pređe kao ni da se zada korak 0.

Komanda RENUM ima format

RENUM [[<novi broj>][,[<stari broj>][,<uvećanje>]]].

(Novi broj) je prvi broj programskog reda koji se koristi u novoj numeraciji. Ukoliko nije naveden podrazumeva se broj 10. (Stari broj) je broj reda od kojeg počinje prenumeracija. Ako je izostavljen podrazumeva se da je to prva linija programa. (Uvećanje) je korak za koji će se uvećati brojevi programskih redova u novoj sekvenci. Ako ga nema takođe se smatra da je 10.

Napominjemo da za razliku od odgovarajuće komande nekih drugih bejzika RENUM osim brojeva redova menja i brojeve koji slede iza naredbi GOTO GOSUB THEN ON ... GOTO ON ... GOSUB i ERL tako da ukazuju na novonumerisane redove. RENUM ne može da promeni redosled linija ili da prenumeriše brojeve redova veće od 65529. Ako želite da promenite redosled više programskih linija bez njihovog ponovnog ispisivanja treba da postupite na sledeći način. Najpre snimite ceo program na primer pod imenom PO. Zatim iz verzije koja je u memoriji komandom DELETE (ne tipkom DELETE već komandom otkucanom slovo po slovo) izbacite sve što sledi iza linija koja je poslednja u sekvenci koje će ostati sa postojaćim bpr ovima i snimite ga pod imenom P1. Učitajte potom staru verziju programa izbacite iz nje sa DELETE sve linije pre i posle onih koje premeštate i snimite ga pod imenom P2. Zatim ponovo učitajte PO i izbacite sa DELETE sve linije koje ste već upamtili kao P1 i P2 i snimite ovaj ostatak pod imenom P3. Učitajte sada P1 i povežite ga pomoću MERGE sa P2 i P3. Razume se ovaj postupak ima smisla samo ako se radi o većem broju programskih linija i obavezno je da sve programe čuvate u aski formatu.

Komanda DELETE ima format

DELETE ((broj linije 1)][−(broj linije 2)]. Slično kao što LIST prikazuje linije iz zadatog opsega tako ih DELETE briše. Jedino što za razliku od LIST ovde ne možete tražiti da se obrišu sve linije počev od neke sa zadatim brojem. Iz sigurnosnih razloga uvek morate navesti broj poslednje programske linije koje se briše. Ukoliko se ne navede ni jedan broj linije ispisuje se poruka Illegal function call jer za brisanje celokupnog programa iz memorije imate na raspolaganju NEW.

Drugi načini

Iz dekadnog brojnog sistema možete prevoditi u binarni i na druge načine na primer metodom oduzimanja stepena broja dva. Recimo broj 720.625 u binarnom brojnom sistemu je 1011010000.101 zato što je



$720=512+128+64+16=2^9+2^7+2^6+2^4$
$0.625=.5+.125=2^{-1}+2^{-3}$

Ovaj način je vrlo zgodan za prevođenje „napamet". U prvo vreme trebaće vam tablica stepena broja 2.

N	2^N	2^{-N}
0	1	1.0
1	2	0.5
2	4	0.25
3	8	0.125
4	16	0.0625
5	32	0.03125
6	64	0.015625
7	128	0.0078125
8	256	0.00390625
9	512	0.001953125
10	1024	0.0009765625

Dekadni zapis stepena broja 2

Još bolji način za prevođenje dekadnih brojeva u binarni brojni sistem je korišćenje posrednog brojnog sistema osnove 2^K. Sistemi računanja čija je osnova ceo stepen broja 2 naročito su interesantni jer se njihov prevod u binarni sistem i kodirani zapis binarnim sistemom podudaraju. Pri tome je stepen broja K ujedno i broj binarnih cifara kojima se svaka cifra datog sistema može kodirati.

Kodovi oktalnih cifara	Kodovi heksadekadnih cifara	
0=000	0=0000	8=1000
1=001	1=0001	9=1001
2=010	2=0010	A=1010
3=011	3=0011	B=1011
4=100	4=0100	C=1100
5=101	5=0101	D=1101
6=110	6=0110	E=1110
7=111	7=0111	F=1111

Tablica binarnih kodova oktalnih i heksadekadnih cifara

Funkcija PEEK koju smo već upoznali daće nam bolji uvid u zadržaj memorija ako se kombinuje sa funkcijama OCT$ i HEX$. OCT$ i HEX$ za dati ceo dekadni broj daju njegov oktalni odnosno heksadekadni zapis pa uz tablicu kodova možete sa lakoćom doći do odgovarajućeg binarnog zapisa. Na primer ? OCT$(720) daje 1320 što bi binarno iznosilo 001 011 010 000 odnosno 1011010000. ?HEX$(720) daje 2D0 što binarno iznosi 0010 1101 0000 odnosno takođe 1011010000.

Iskoristite sada sve ove informacije da odgovorite na pitanje koliko je 157 OR 202, 157 AND 202, 157 IMP 202 i 157 EQV 202. Proverite svoje odgovore zatraživši od računara da vam otkuca rezultate.

Žvrljanje po memoriji

Na kraju upoznajmo vrlo opasnu naredbu POKE. Ona je opasna kada se neoprezno upotrebljava jer direktno menja sadržaje memorijskih registara pa tako može ne samo da upropastite sopstveni program već i da oborite interpretator ili operativni sistem. Zato koristite POKE samo ako ste sigurni u to šta radite.

Proverite dejstvo sledećeg programa

```
10 A%=10
20 ADR=VARPTR(A%)
30 POKE ADR 1  POKE ADR+1 1
40 PRINT A%
```

Vidite da ste umesto 10, koliko ste dodelili promenljivoj A% dobili 257 tj 1+256·1 što smo sa POKE stavili na memorijske lokacije gde se čuva A%.

Proverite gde se u memoriji čuvaju podaci otkucavši ? ADR i koristite odgovarajuću memorijsku lokaciju za svoje eksperimente sa POKE. Kada bolje upoznate svoj računar i raspodelu njegovog memorijskog prostora moći ćete da koristite POKE kao vezu između bejzika i mašinskih programa.

Šesti susret

Proširite rečnik

Kada govorite ili pišete, koristite stotine hiljada reči na računajući tu lična imena. Neka od njih su u upotrebi vekovima i svi ih stalno koriste. Druge, na primer stručne računarske termine, koriste samo ljudi odgovarajuće profesije. Za razliku od govornog, u jeziku struke ima malo reči, ali njima može puno da se kaže. Lingvisti tvrde da se sa fondom od samo 3.000 reči engleskog jezika može razumeti 80% onoga što je ikada napisano na tom jeziku. Bejzik je znatno ograničeniji od engleskog ili ma kog drugog prirodnog jezika. Za početak, on se sastoji od svega stotinak reči. Videli ste da se mogu sastaviti vrlo složeni programi, a da se ne upotrebi ni pola tog fonda. Proširite svoj bejzik rečnik novim funkcijama i naredbama.

Dve osnovna operacija koja se vrše nad znakovnim podacima su spajanje i razdvajanje. Spajanje se u bejziku realizuje operatorom + a za razdvajanje nije predviđen poseban operator već se ostvaruje pomoću funkcija LEFT$ MID$ i RIGHT$. Za rad se znakovnim podacima koristi se i naredba MID$. Uz funkcije HEX$, OCT$, INKEY$, STRING$, ASC i CHR$ koje smo već predstavili, u ovoj verziji bejzika na raspolaganju su i funkcije LEN, VAL, INSTR, STR$, INPUT$ i SPACE$.

Podsećamo da je ASC numerička funkcija koja ima format ASC (<string>) i proizvodi aski vrednost prvog znaka iz niske <string>. Funkcije CHR$ ima format CHR$ (<brojni izraz>) i proizvodi znak čija je aski vrednost jednaka argumentu funkcije. OCT$ i HEX$ za dati ceo dekadni broj daju njegov oktalni odnosno heksadekadni ekvivalent. Format prve je OCT$(X) i one daje grupu znakova koja predstavlja zapis celog broja X u oktalnom brojnom sistemu. Ako X nije ceo broj, biće zaokružen. Na odgovarajući način radi i HEX$(x). Funkciju STRING$(<a> <b>) pominjali smo kada smo poredili mogućnosti ispisa naredbi PRINT i TEXT. Ona proizvodi reč koja se sastoji od <a> znakova koji imaju aski kod <b>. Slično funkcija SPACE$(X) može postići i upotrebom STRING$(X 32).

Izdvajanje delova reči

Funkcije LEFT$, RIGHT$ i MID$ omogućavaju izdvajanja delova reči. Funkcija LEFT$ ima format

LEFT$(<string> <broj>)

Ova funkcija izdvaja prvih <broj> levih znakova iz podatka <string>. Argument (broj) mora biti ceo broj iz intervala [0 255]. Ako je taj broj veći od dužina reči, uzima se ceo podatak, a ako je 0 uzima se prazna reč. Na primer LEFT$(„ANANAS" 3) daje „ANA".

Funkcija MID$ ima format MID$(<string> <broj1>[,<broj2>]). Ona izdvaja podnisku dužina <broj 2> iz reči <striog> počev od pozicija <broj 1>. Dozvoljene vrednosti za <broj 1> su između 1 i 255, a ako se navede broj van ovog intervala dobija se poruka Illegal argument. <Broj 2> može biti iz intervala [0.255]. Ako je prvi brojni argument veći od dužina <stringa>, rezultat je prazna reč, a ako se izostavi drugi argument uzima se sve što preostane. Na primer MID$(„BANANA" 2,3) daje „ANA". Primetimo da je MID$(S$ 1)<=>LEFT$(S$,I)

Funkcije RIGHT$ ima format RIGHT$(<string> <broj>). Ona izdvaja podreč koja se sastoji od desnih <broj> znakova znakovnog podatka <string>. Argument <broj> mora biti iz intervala [0.255]. Ako je 0 dobija se prazna reč.

Predstavimo sada funkciju koja omogućava sastavljanja programa koji daje uvid u dejstvo prethodne tri funkcije. To je funkcija LEN čiji je format LEN(<string>). Ona izračunava broj znakova u podatku. Blanko pozicija i negrafički simboli se takođe uračunavaju. Na primer LEN(„ANANAS") daje 6, LEN(„A & B") daje 5, LEN(CHR$(13)) daje 1.

```
10 A$=1234567890
20 L=LEN(A$)
30 FORI=1TO L
40 PRINT LEFT$(A$ I)+RIGHT$(A$ L-I)
50 NEXT
```

Program se slike štampa niz kopija reči A$ koje stvara kao zbir levih I i desnih LEN(A$)−I znakova, pri čemu I uzima vrednosti od 1 do LEN(A$). Još bolji uvid u dejstvo funkcija LEFT$ i RIGHT$ možete postići ako u prethodnom programu modifikujete naredbu PRINT na sledeći način

```
PRINT LEFT$(A$,I) RIGHT$(A$ L-I)
```

jer će se tako delova reči A$ koje proizvode LEFT$ i RIGHT$ štampati razdvojeno.

U našoj verziji bejzika sem funkcije MID$ postoji i naredba se istim imenom. Njeno dejstvo je zamena grupe znakova u azbučnom izrazu nekom drugom grupom znakova. Naredba MID$ ima format

MID$(<st1> <p>[<k>]=<st2>

gde su <st1> i <st2> znakovni a <p> i <k> celobrojni izrazi. Znakovi u <st1> zamenjuju se počev od p-tog znakovima iz <st2>. Pri tom se menja k znakova. Ako treći parametar nije naveden <st2> će se prepisati do dužina <st1>. Kako radi ova naredba ilustruje sledeći program

```
10 CLS: PRINT: PRINT TAB(30)"DEJSTVO NAR
EDBE MID$"
20 PRINT:PRINT"ZA NASTAVAK PRITISNITE BI
LO KOJU TIPKU"
30 PRINT:PRINT"OD POZICIJE I ZAMENJUJE S
E J ZNAKOVA": PRINT
40 PRINT"REC"," I"," J"
50 A$="1234567890"
60 FOR I=1 TO 11
70 A$="1234567890"
80 FOR J=0 TO 11
90 MID$(A$,I,J)="AAAAA"
100 PRINT A$,I,J
110 INPUT X$
120 NEXT J
130 NEXT I
```

Listing 15

Funkcije konverzije

U programima je često potrebno cifre zabeležena u aski formatu prevesti u brojni podatak ili niz cifara nakog broja pretvoriti u znakovni podatak. Ovo omogućavaju funkcije konverzije STR$ i VAL. Prva ima format STR$(<broj>) a druga VAL(<string>).

STR$ daje azbučnu reprezentaciju argumenta te se koristi prilikom formatizacija štampe. Na primer PRINT STR$ (1234500000000) i PRINT STR$ (.000000000000001) će proizvesti 1.2345E+12 i 1E−15. Imajte u vidu da STR$ ubacuje vodeće blanko praznine pa bi LEN(STR$(1)) dalo vrednost 2.

VAL izračunava numeričku vrednost koja je data <string> argumentom. Ako prvi karakter nije +, −, blanko ili cifra odgovarajuća vrednost je 0. Konverzija znakovnog podatka završava se nailaskom na znak koji nije cifra (ili decimalna tačka ili oznaka E za exp.). Stoga VAL(„7 PATULJKA") daje vrednost 7, a VAL(„SNEŽANA i 7 PATULJAKA") daje vrednost 0.

Pretraživanje teksta

Pretraživanje teksta i zamena odgovarajuće reči ili simbola nakom drugom reči ili simbolom počiva pored naredbe MID$ i na funkciji INSTR. Ova funkcija ima format INSTR([I]X$ Y$). Njeno dejstvo je vraćanje pozicija od koje se Y$ pojavljuje u X$. Opcija I je pozicija od koje će se pretraživati, a ako nije navedeno smatra se da je 1. I mora biti u opsegu 1 do 255, u suprotnom se daje poruka greške Illegal argument. Ako je I>LEN(X$), X$ prazna reč ili Y$ ne postoji INSTR će vratiti vrednost 0. Program

```
10 X$="UČITI UČITI I SAMO UČITI"
20 PRINT INSTR(X$ "UČITI")
```

štampa vrednost 1. Ako liniju 20 zamenimo sa

```
20 PRINT INSTR(10 X$, "UČITI")
```

dobićemo 21.

Korišćenje lozinki

Uz mogućnost korišćenja programa čije su izmene i listanje zabranjeni, „timov" bejzik nudi i mogućnost da određena programe ili datoteke mogu da upotrebljavaju samo korisnici koji znaju odgovarajuću lozinku. To omogućava funkcija INPUT$ čiji je format INPUT$(X[#]Y). Ona vraća grupu znakova dužina X pročitanu sa tastature ili iz datoteke pod brojem Y. Ako je za ulaz iskorišćena tastatura, znaci koji se kucaju na prikazuju se na ekranu. Svi znaci mogu biti rezultat funkcije INPUT$ osim CTRL C koji prekida rad programa.

Program za prevođenje binarnih u dekadne brojeve prikazan na listingu 16 koristi izložena mogućnosti rada se znakovnim promenljivim. Za ilustraciju upotrebe funkcije INPUT$ umetnut je potprogram od linije 300 koji se poziva na samom početku rada. On na sredini ekrana ispisuje poruku kojom traži da se unese šifra. Ako se šifra korektno otkuca, upravljanje se vraća u program, a ako ne, prekida se sa izvršavanjem programa. Funkcija INPUT$ u ovom slučaju prihvata prvih sedam znakova otkucanih na tastaturi i dodeljuje ih promenljivoj S$. U našem programu šifra je „TIM—011". Za razliku od obične naredbe ulaza pri upotrebi INPUT$ na ekranu se ne pojavljuju znaci koji se kucaju.

Naredba INPUT$ primenjeno u programu koji je snimljen sa nastavkom P zaista može da se upotrebi kao sredstvo koje sprečava neovlašćeno korišćenje programa, jer korisnik tako snimljen program ne može ni da izlista ni da izmeni. Svaki pokušaj listanja i izmena programa izaziva pojavu poruke greške *Illegal function call*. Ni ideja da se takav program snimi na uobičajeni način pa onda učita i izlista ne može da se realizuje. Takav pokušaj završava se istom porukom neuspeha.

U našem programu se, po razrešenju dileme da li korisnik ima pravo da ga upotrebljava, postavlja zahtev da se unese binarni broj. Unet broj se prihvata kao znakovna promenljiva B$ kojoj se prvo određuje dužina, eliminiše vodeći simbol znaka, ako postoji, i određuje mesto pozicione tačke u zapisu. Za to su upotrebljena funkcija LEN, LEFT$, RIGHT$ i INSTR.

Potom se, po ranije opisanom algoritmu, izračunava vrednost dekadnog ekvivalenta koja je u programu označena sa B. Znak broja se čuva u promenljivoj Z. Za izdvajanje pojedinih cifara binarnog broja korišćena je funkcija MID$. U slučaju da se pri tom naišlo na neki znak različit

od 0' i 1 program bi zahtevao unos novog binernog broja. Na kraju se za izdavenje rezultata koristi funkcija STR$

```
10 'PREVODJENJE @INARNI@ PROJEVA U DEKADNE
20 CLS
30 GOSUB 300
40 CLS
50 INPUT "UNESITE @INARNI @ROJ",B$
60 '***ODREDJIVANJE DUZINE ZAPISA
70 D=LEN(@$)
80 '***UTVRDJIVANJE ZNAKA @ROJA
90 IF LEFT$(B$,1)="-" THEN Z=-1 ELSE Z=I
100 '***ELIMINISANJE ZNAKA BROJA IZ ZAPISA
110 IF LEFT$(@$ 1)="-" OR LEFT$(B$,1)="+"
THEN @$=RIGHT$(B$,D-1) D=D-1
120 'UTVRDJIVANJE MESTA POZICIONE TACKE
130 P=INSTR(B$,".")
140 '*** RACUNANJE CELOG DELA BROJA
150 IF P=0 THEN G=D ELSE G=P-I
160 B=0
170 FOR I=1 TO G
180 C$=MID$(B$,I,1)
190 IF C$<>"0" AND C$<>"1" THEN 50
200 IF C$="1" THEN B=B+2^(G-I)
210 NEXT I
220 '***RACUNANJE RAZLOMLJENOG DELA @ROJA
230 FOR I=G+2 TO D
240 C$=MID$(B$,I,1)
250 IF C$<>"1" AND C$<>"0" THEN 50
260 IF C$="1" THEN B=B+2^(F-I)
270 NEXT I
280 PRINT STR$(B*Z)
290 END
300 TEXT "OTKUCAJTE SIFRU",220,100,0,3
310 S$=INPUT$(7)
320 IF S$<>"TIM 011" THEN CLS: END
330 RETURN
```

Listing 16

Numeričke funkcije

Oni koji pišu programe za razne finansijska izračunavanje obično ne žele da se na ekranu ili štampaču pojavljuju negativni brojevi. Možda recimo program koji vodi evidenciju o stanju tekućeg računa koristi i negativne brojeve ali prilikom listanja izdatih čekova odgovarajuće vrednosti štampa bez minusa. Negativni brojevi su nepoželjni i pri izračunavanju nekih funkcija. Funkcije SOR koja računa vrednost kvadratnog korena i LOG koja računa vrednost logaritma za osnovu e nisu definisane za negativne argumante. Korišćenjem funkcije ABS može se sprečiti pojava argumenta za koje ove funkcije nisu definisane kad i štampanje negativnih vrednosti kada ih ne želimo. Funkcija ABS koja daje apsolutnu vrednost broja može se korisno upotrebiti i na mnoge druge načine. Recimo ako treba da sastavimo potprogram koji po pozivu promenljivoj T, naizmenice dodeljuje vrednosti 3 i 16 to možemo bez funkcije ABS rešiti na sledeći način

```
t T=3
500 IF T=3 THEN T=16  RETURN
510 IF T=16 THEN T=3  RETURN
```

Korišćenjem ABS ovaj problem se rešava mnogo elegantnije

```
500 T=ABS (T-19)  RETURN
```

Ako je prethodna vrednost za T bila 3 onda će linija 500 po pravilu

T=ABS (3-19)=ABS (−16)=16

Promenljivoj T dodeliti vrednost 16. Ako je pak bila 16 po pravilu

T=ABS(16-19)=ABS(16)=3

će je nanovo promeniti u 3

Sa znakom broja radi i funkcija SGN. Po definiciji SGN(X) iznosi i ako je X broj veći od 0 0 ako je broj X jednak 0 i −1 ako je X negativan. Funkcija SGN omogućava da se i u bejziku realizuje naredbe prenosa upravljanja po vrednosti aritmetičkog izraza koje postoji u fortranu izrazom

ON SGN(X)+2 GOTO 100 200 300

Predaje se upravljenje redu 100 ako je X negativno redu 200 ako je X nula a redu 300 ako je veće od nule

Zaokruživanje brojeva

Većina bejzika za zaokruživanje brojeva koristi samo funkciju INT. Ona odbacuje sve cifre razlomljenog dela pozitivnog broja pa je INT(0 1)=0 ali i INT(0 99)=0. Za negativne vrednosti, funkcija INT zaokružuje na prvi manji negativan broj na primer INT(−0 1)=−1. Uobičajan način zaokruživanja na bliži ceo broj može se simulirati pomoću izraza INT(X+5). Tako će INT(0 1+0 5) dati 0 INT(1 9+0 5) t a INT(0 1+0 5) 0 U timovom bejziku nema potrebe za simuliranjem uobičajenog načina zaokruživanja jer postoji funkcija CINT koja upravo to radi

Celobrojni deo bliži nuli može se odrađivati pomoću izraza SGN (X)*INT(ABS(X)). Za X=1 9 ovaj izraz daje vrednost 1 a za X=1 9 daje −1. Ni ovaj izraz ne morate pisati kada vam je potreban celobrojni deo bliži nuli jer naš bejzik ima funkciju FIX koja daje upravo takvu vrednost

Slučajni brojevi

Svi kućni računari poseduju generatore slučajnih brojeva koji se iz bejzika pozivaju pomoću funkcije RND (RaNDom=slučajan). Ova funkcija proizvodi jedan slučajan broj iz otvorenog intervala (0 1). Njeno dejstvo može se videti iz sledećeg kratkog programa.

```
10 X=RND  PRINT X  GOTO 10
```

Po startovanju on počinje da štampa niz decimalnih brojeva 245121 305001 Ako se program prekine i ponovo startuje isti niz brojeva ponovo promiče ekranom, što znači da oni uopšte nisu slučajni. Ako hoćemo da budemo precizni trebalo bi da kažemo da računari koriste pseudoslučajne brojeve koji samo gledano spolja nisu međusobno zavisni. Konačno postupak koji nije slučajan i ne može proizvesti prava slučajne brojeve. Međutim za korisnika koji ne zna po kom se algoritmu proizvode oni su dovoljno nepoznati da bi mogao da ih upotrebljava kao slučajne

Puno knjiga posvećeno je oštroumnoj tehnici koja slučajne brojeve koristi za rešavanja složenih zadataka numeričke analize. Generisanje slučajnih vrednosti omogućava da se na računarima modeliraju prirodne pojave a predstavlja i dobar izvor podataka pri testiranju različitih algoritama. Stoga je sposobnost računara da izmišlja slučajne brojeva značajnija funkcija nego što se često misli

U timovom bejziku generišu se slučajni brojevi iz unapred određenog niza sa 65536 članova. Prvi od njih tzv random sid (random seed) je broj čijih se cifara po nekom algoritmu dobija sledeći slučajni broj. Nad ciframa tako dobijenog broja ponavlja isti postupak za dobijanje trećeg i tako redom. Na našem računaru prvi slučajni broj je 308601

Uz funkciju RND može se napisati i argument između zagrada. Ako je argument negativan broj dobije se uvek prvi slučajan broj. Ako je jednak nuli ponavlja se prethodni slučajni broj a ako je pozitivan dobija se ista sekvenca kao kada nije naveden argument. Da se uvek ne bi ponavljala ista sekvenca slučajnih brojeva kao što bi se desilo u jednom od naših prethodnih programa da smo umesto GOTO na početak programa upotrebili RUN, koristi se naredba RANDOMIZE. Njena namena je zadavanje random sida. Ako se RANDOMIZE upotrebi bez navođenja celobrojnog izraza proizvodi pitanje

Radnom number seed (−32768 to 32767)?

pa korisnik može unošenjem raznih brojeva da niz pseudoslučajnih brojeva učini slučajnijim

U praksi su često potrebni celi brojevi iz nekog intervala recimo brojevi od 1 do 39 ako želimo da simuliramo igru loto. U tom slučaju koristimo neku od funkcija zaokruživanja recimo INT INT (RND*B+P) daje neki od B celih brojeva počev od P. Konkretno INT (RND*39+1) proizvodi neki ceo broj iz intervala [1,39]

Formatizovanje izlaza

Pod formatizovanim izlazom podrazumeva se sređeno pozicioniranje komentara i podataka koje želimo da učinimo vidljivim na ekranu prema određenim kriterijumima. Većina bejzika za tu svrhu raspolaže funkcijama SPC TAB i POS. SPC(I) štampa I blanko znakova i može se upotrebiti samo uz PRINT i LPRINT naredbu za razliku od SPACE$ koja se može koristiti samostalno. Uz SPC se podrazumeva znak tačka-zarez na kraju. Funkcija POS mora se pisati u formatu POS(I) mada sam argument ništa specijalno ne znači. Ona vraća poziciju kursora u okviru linije pri čemu se podrazumeva da su pozicije na ekranu numerisane od 1 do 80. Slično je i dejstvo funkcija LPOS(I) koja daje poziciju glave štampača u okviru linije koja se štampa. Na primer, naredba

IF POS(O)>75 THEN PRINT CHR$(13)

zadaje desnu marginu teksta na 75 koloni tj prenosi štampu u sledeći red. Kada se odštampa prvih 75 kolona

Funkcija TAB(I) odgovara tabulaciji na pisaćoj mašini. Ona se može upotrebljavati jedino u okviru PRINT i LPRIN naredbe a proizvodi štampanje počev od I-te kolone. Ako je I manje od trenutne pozicije kursora štampa se u sledećem redu počev od I-te pozicije. Kod nekih drugih računara u ovom slučaju se ignoriše TAB. Za argument I je u memoriji rezervisan samo jedan bajt te on može imati vrednosti do 255. U našoj verziji bejzika TAB(O) ima isto dejstvo kao TAB(1). Program

```
10 I$ MILENA
20 FOR I=1 TO 255
30 PRINT TAB(I)I$
40 NEXT I
```

ilustruje dejstvo funkcija TAB. Primetimo da odsustvo znaka razdvajanja između TAB i I$ interpretatoru ne smeta da izvrši naredbu 30. Valja upamtiti da se podatak koji je duži od prostora koji mu do kraja reda ostavlja TAB štampa od početaka sledećeg reda. Tako je u našem programu za vrednosti ciklusne promenljive 76 77 78 79 i 80 ime MILENA ' štampano od početaka narednog reda

Sem ovih standardnih mogućnosti bejzika za formatizaciju štampa u priručniku možete videti i PRINT USING. Na žalost u verzijama interpretatora koje su na početku podeljene školama ova naredba ne radi korektno. Greška je kasnije ispravljena pa ako imate staru verziju sistemske diskete iskoristite neki od programa za kopiranje i presnimite modifikovan interpretator. Najjednostavnije ćete rešiti problem uslužnim programom DCP koji kopira celokupni sadržaj diskete, ali prethodno sačuvajte na pomoćnoj disketi programe koje ste sami kreirali

Sedmi susret

Računanje na računaru

Matematika je oduvek bila bauk za prosečnog đaka jer je zahtevala puno razmišljanja, koncentracije i mukotrpnih računanja. Prepuštanjem procesa računanja računaru omogućuje



nam da se više posvetimo razumevanju ideja matematike. Funkcija n! često se koristi u matematici. Po definiciji, n! predstavlja proizvod svih prirodnih brojeva do n (0!=1). Nije neki logički problem izračunati vrednost 10!, ali ima posla. Stoga bi program za računanje faktorijala dobro došao. Ako se držimo samo definicije, program bi mogao ovako da izgleda.

```
[illegible] PROGR[illegible]
[illegible]
[illegible] PROGRAM
[illegible] THEN RETURN
[illegible]
```

Ako testirate ovaj program utvrdićete da se na svakom računaru on završava porukom o prekoračenju ako unesete vrednost veću od 33. Neregularan završetak nastupa i onda kada se unese podatak manji od 0 ili broj koji nije ceo. Na žalost, u tim slučajevima program ne prijavljuje grešku, dakle nije korektan. Funkcija n! je definisana samo za prirodne brojeve i nulu i program bi morao da spreči poziv potprograma za računanje n! sa drugačijim argumentima. Relativno lako možemo da sprečimo te neregularne situacije. Između linija 20 i 30 treba umetnuti kontrolu ulaznih podataka.

```
12 IF N< THEN PRINT "N! NIJE DEFINISANO ZA NEGATIVNE BROJEVE": GOTO 20
[illegible] SAMO [illegible]
16 IF N[illegible]
OVOG BRO[illegible]
MENE [illegible]
```

Umesto linija 16 mogli smo koristiti naredbu ON ERROR GOTO.

Programska kontrola grešaka

Ako želimo da greške obrađujemo u programu, kao u našem slučaju prekoračenja, onda naredbom ON ERROR GOTO (bpr) prenosimo upravljanje na segment koji obrađuje greške. Dve promenljive sa rezervisanim imenima ERR i ERL omogućavaju programsku kontrolu grešaka. Kada nastupi greška, u promenljivu ERR se upisuje kod greške, a u ERL broj linije u kojoj je greška nastupila. Kodove grešaka možemo videti u prilogu poruke o greškama, a možemo i pomoću naredbe ERROR koja simulira pojavu greške. Recimo, ERROR 1 otkucano u direktnom režimu (bez naredbe PRINT) daje poruku Syntax error, znači kôd sintaksne greške je 1. ERROR 6 proizvodi poruku Overflow — prekoračenje.

U našem programu bi programska kontrola greška prekoračenja mogla da se realizuje dodavanjem sledećih linija.

```
[illegible] ON ERROR G[illegible]
[illegible] IF ERR [illegible]
OVOG BROJA POSTO[illegible] AL[illegible] E PREVELIK ZA
MENE RUN
```

Program ne mora, kako smo to predvideli u liniji 200, da se izvršava iz početka po obradi greške. Nastavak rada može biti i od naredbe koja je prouzrokovala grešku ili naredbe koja sledi neposredno za njom. Ovo omogućava naredba RESUME koja se može koristiti u četiri formata

```
RESUME
RESUME 0
RESUME NEXT
RESUME bp
```

U prva dva formata ponavlja se naredba koja je prouzrokovala grešku. RESUME NEXT nastavlja izvršavanje programa od prve naredbe posle one koja je izazvala grešku, a RESUME <bpr> prenosi upravljanje na liniju sa odgovarajućim brojem programskog reda.

Uz korišćenje RESUME, linija 200 našeg programa glasi

```
200 IFF ERR=6 TEN PRINT "FAKTORIJAL OVOG BROJA POSTOJI ALI JE PREVELIK ZA MENE": RESUME
```

Kako povećati tačnost

Drugi, mnogo veći problem od kontrole ulaznih podataka je tačnost računanja. Ovako kako smo mi uveli promenljivu F, faktorijel brojeva do 33 se računa samo sa 6 značajnih cifara. Tako on za 20! daje vrednost 2.4329E+18. Izmenimo li ime promenljive F u F#, tj. deklarišemo li je kao promenljivu dvostruke preciznosti, za 20! dobijamo vrednost 2.432902000-817664D+18. Međutim, poučeni primerom računanja vrednosti 1/3 u dvostrukoj tačnosti moramo ovaj drugi podatak prihvatiti sa rezervom i za svaki slučaj, ako želimo tačnija računanja, rešiti problem po sasvim drugom algoritmu. (Cifre tačno izračunatog faktorijela broja n sačuvane kao elemente niza treba redom množiti sa <n+1> i tako dobiti tačnu vrednost <n+1>! Ovakva rešenja možete potražiti u zbirkama zadataka iz programiranja.)

Primena dvostruke tačnosti povećava broj značajnih cifra rezultata, ali ne omogućava da izračunamo vrednost 34!. To doduše možemo i bez uvođenja nizova, ako nekim programerskim "trikom" eliminišemo mnoštvo nula s kraja broja n!.

Prepravkom linija 60 u

```
60 PRINT F# "10t" S
125 IF INT(F#/10)=F#/10 THEN S=S+1: F#=F#/10: GOTO 125
```

dobijamo približne vrednosti za faktorijale mnogo većih brojeva. Recimo, primenom tako modifikovanog programa saznajemo da je 100! broj reda veličine 10↑157.

a kako brzinu računanja

Modifikovan program za računanje faktorijela i korišćenja dvostruke preciznosti dali su nam uvid u brzinu rasta funkcije n!. Međutim, i prva verzija programa davala je informaciju koju vredi upamtiti, a to je da je 10!=3628800, približno 3.5 miliona. U neku ruku 10! predstavlja granicu između onoga što možemo i što ne možemo sračunati na računaru. Naime, ako algoritam zahteva proveru više od 10! slučajeva i ako svaka provera zahteva po jednu milisekundu mašinskog vremena, to bi značilo da je računaru potrebno više od sat vremena za proveru tih uslova. Stoga valja dobro razmisliti o dužini rada programa i imati na umu da nepotrebne instrukcije umetnute u petlju višestruko rasipaju mašinsko vreme.

Kao što su male dosetke mogle da povećaju tačnost računanja, tako mogu da uštede i mnogo mašinskog vremena. Uporedite sledeće dva programa koji proizvode istovetne rezultate

```
1 PROGRAM 1
[illegible] FOR I=1 TO [illegible]
[illegible] A[illegible] 0
[illegible] FOR J=1 TO [illegible]
40 [illegible]
50 NEXT J
60 NEXT I
1 PROGRAM 2
10 FOR I=1 TO 100
20 B=0.0
30 FOR J=1 TO 50
40 B=B+Z(I,J)
50 NEXT J
60 A<I>=X*Y*B
70 NEXT I
```

Pri računanju na drugi način ne štedi se samo 4999 množenja već i 5000 računanja adrese A(I).

NASTAVAK NA 35 STRANI

ELEKTRONSKA NASTAVA

CLASSNET

Koja je uloga predavača u društvu sa dinamičnim tehnološkim razvojem?

To nije više samo prenošenje znanja sistema vrednosti i verovanja koja će pomoći primaocima da nađu svoje mesto u društvu koje se neznatno menja u toku njihovog životnog veka

Tehnologija se danas razvija tako brzo da verovatno svaka osoba mora da revidira veći deo stečenog znanja novim, nekoliko puta u toku svog radnog veka

Na sreću danas se zna mnogo više o metodama prenošenja znanja, a tehnološki razvoj obezbeđuje sredstva kojima se ona mogu preneti brže većem broju primaoca nego što je to bio slučaj ranije

Uloga predavača radikalno se menja jer on mora da ovlada novim didaktičkim sredstvima mora naučiti kako da se njima služi i kako da u potpunosti iskoristi njihov potencijal Kod svojih učenika treba da razvija kreativnost fleksibilnost sposobnost prijema novih znanja sposobnost analiziranja primljenih informacija i njihovog organizovanja

Lični uticaj predavača koji je neophodan kao podsticaj svakom učeniku ne može se ničim zameniti Međutim postoje sredstva koja pomažu predavaču u razvoju novih nastavnih metoda Jedno od Olivetti-jevih rešenja u ovoj oblasti je CLASSNET To je softverski paket podržan lokalnom mrežom (LAN) preko koje se vrši povezivanje personalnih računara pod operativnim sistemom MS-DOS CLASSNET ili elektronska učionica sastoji se iz nekoliko nivoa koji omogućavaju deobu resursa i perifernih uređaja, kao i istovremeni pristup eksternim memorijama sa više radnih mesta Radne stanice predavača i učenika mogu međusobno da komuniciraju

CLASSNET ima dva načina rada — obična učionica i elektronska tabla Kod prvog predavač može da pokupi i prikaže na svom ekranu sadržaj svakog učenika da ga izmeni memoriše ili vrati odgovarajućem učeniku Moguća je i međusobna razmena poruka Kod elektronske table tekst koji predavač ispisuje na svom ekranu prikazuje se na ekranima učenika a predavač može u bilo kom trenutku da „preda kredu" nekom od učenika sa zahtevom da on nastavi rad

Predavač može ako želi da pušta i neki aplikativni program sa sadržajem iz matematike fizike hemije ekonomije i slično

Celokupan rad u toku predavanja može se memorisati što omogućava predavaču da prati rad svakog učenika

CLASSNET ima takvu arhitekturu koja dozvoljava proširenje i povezivanje sa drugim sistemima

Ukoliko postoji zahtev spremni smo da organizujemo demonstraciju rada navedene opreme

Za sve dodatne informacije stoje Vam na raspolaganju naši specijalisti

R O „DINARA
SEKTOR INFORMATIKA
Ul Vlajkovićeva br 5
11000 BEOGRAD
tel. 011/ 335-886,
335-887



NASTAVAK SA 33. STRANE

jer se pri svakom obraćanju elementu niza treba izračunati apsolutna adresa elementa niza na osnovu početne adrese niza, broja registara koje zauzima svaki element i konkretne vrednosti indeksa.

Uz to, uvek kad množenje možete zameniti sabiranjem ili stepenovanje množenjem, učinite to. Ne samo da ćete brže doći do rezultata nego imate i šansu da dođete do tačnog rezultata. Da je 15·15·15·15·15=759375 možete izračunati i bez računara. Rezultat u svakom slučaju mora da se završava na 5. Ako međutim zatražite od „tima" da ovo izračuna sa ?15^5 dobićete vrednost 759376. Znači, nije u pitanju greška na 6. decimali, ni na trećoj, greška je u celom delu broja! Ako istu stvar zahtevate preko množenja, dobićete tačan rezultat.

Primetili ste, verovatno, da nismo posvetili dužnu pažnju funkcijama EXP, LOG, SIN i još nekim koje su opisane u priručniku. Istraživanje koliko su odgovarajuća računanja tačne i brza u GBASICu ostavljamo vama. Činilo nam se da je korisnije istaći kvalitete ove verzije bejzika, a to su pre svega odlična grafika i velike mogućnosti rada sa znakovnim podacima. Zato, na kraju, samo još nekoliko reči o dve grafičke naredbe koje nismo opisali i korišćenju zvuka.

Bojenje crteža

Za popunjavanje zatvorenih površi određenim intenzitetom možemo koristiti dve naredbe koje na prvi pogled imaju slično dejstvo. To su naredbe FILL, koja popunjava površine do granice omeđene zadatim intenzitetom i PAINT koja menja boju površine. Obe naredbe treba pisati sa standardne četiri parametra od kojih su prva dva obavezna. Razliku u njihovom dejstvu ilustruje program na listingu 17 koji je sastavio učenik Saša Malkov.

```
1 'MALKOV SASA
2 'RAZLIKA IZMEDJU PAINT I FILL
10 CLS
20 MOVE 256 128
30 ELIPSE 50,100,1
40 ELIPSE 120,50,2
50 TEXT "PAINT BOJI DO GRANICA ZATVORENE LINIJE",0,240,0 3
60 PAINT 0.0 1.3
70 FOR A=1 TO 5000 NEXT A
80 TEXT "FILL BOJI DO LINIJE ZADATOG INTENZITETA",0 200
90 FILL 0,0,1,2
```

Listing 17

PAINT neće ispuniti površinu ako je intenzitet tačke od koje počinje popunjavanje jednak zadatom intenzitetu ili ako je zadata tačka van vidljivog ekrana.

```
1 REM PERSPEKTIVNI PRIKAZ SFERE Ivan Pribicevic
3 INPUT"Uvelicanje (najbolje od 50-1000) d=",D
4 INPUT"POLUPRECNIK (vidljive od 10-255) r=",R
5 INPUT"CENTRALNI UGAO POLIG. POVRSI HORIZ. PRESEKA SFERE (delilac od 180) E=",F
7 CLS P=E S=E K=0 L=E
10 FOR N=-90 TO 90-E STEP P
20 FOR M=0 TO 360 -E STEP S
30 SN=N/180*3 14152
40 PN=(N+K)/180*3 14152
50 SM=M/180*3 14152
60 PM=(M+L)/180*3.14152
70 X=R*SIN(SM)*COS(SN)+R+D
80 XO=R*SIN(PM)*COS(PN)+R+D
90 Y=R*COS(SM)*COS(SN)+R
100 YO=R*COS(PM)*COS(PN)+R
110 Z=R*SIN (SN)+R
120 ZO=R*SIN(PN)+R
130 A=Y*D /X
140 B=(Z*D/X)/2
150 AO = YO *D/XO
160 BO=(ZO*D/XO)/2
170 PLOT A,B DRAW AO,BO
180 NEXT M NEXT N
190 K=E L=0 GOTO 10
```

Listing 18

Kao ilustraciju korišćenja grafičkih mogućnosti „tima" iz bejzika navodimo i program Ivana Pribićevića, učenika 2. razreda, koji među svojim nastavnim predmetima uopšte nema programiranje. On je „kradom memorijskih ciklusa", tj. na odmorima između časova, uspeo da kreira program prikazan na listingu 18. Pogledajte kako ovaj program crta sferu za ulazne veličine 400 200 10.

Korišćenje zvuka

Za korišćenje zvuka imamo na raspolaganju naredbu SOUND. Format naredbe je

SOUND <visina> <trajanje>

Trajanje se zadaje u milisekundama i može imati vrednost do 65535. Parametar visine može imati vrednost 0 do 63. Vrednosti veće od 63 svode se na ovaj opseg po modulu 64. Iz priručnika možete videti kakva značenja imaju vrednosti visine, a iz programa prikazanih na Sl. 19 i Sl. 19a kako se praktično upotrebljava naredba SOUND.

```
5 DIM T$(20)
10 INPUT"UNESI TEMPO 1-3",TEMPO
20 INPUT"UNESI BOJU=0-1";BOJA:BOJA=BOJA*32
30 FOR I=1 TO 20
40 READ T$(I)
50 NEXT I
60 DATA Q,2,W,3,E,R,5,T,6,Y
65 DATA 7,U,I,9,O,0,P,|,+,\
70 A$=INKEY$ IF A$="" THEN 70
80 IF A$="K" THEN STOP
90 I=1
100 IF A$=T$(I) OR I=20 THEN 130
110 I=I+1
120 GOTO 100
130 SOUND BOJA+I+1,100*TEMPO
140 GOTO 70
150 END
```

Listing 19

```
10 INPUT"UNESI TEMPO 1-10";TEMPO
20 INPUT"UNESI BOJU TONA 0-1",BOJA BOJA=BOJA*32
30 S=1
40 WHILE S>0
50   READ S,T
60   IF S>0 THEN SOUND S+BOJA,T*100*TEMPO
70 WEND
80 DATA 7 1,16,1 5,14,0 5,12,1,14 1,12,1,9,1 7 1,4,4,7,1
90 DATA 16,1,5 14,0 5,12,1,12,1,11,1,12,1,14,5
100 DATA 7,1,16,1 5,14,0 5,12,1,14,1,12,1,9,1 7,1,4,4
105 DATA 7,1,9,1 14,1,12 1,11,1,9,1,11,1,12,5,1,1
110 DATA 7,3,12,3,9,3,14,3,11,1 11 1,11,1,11 1,9,1,11,1,12,2,14,1,16,3
120 DATA 7,3,12 3,9,3,14,2 12,1,11,1,11,1 11,1,11,1,9,1 11 1 12 3,-1,-1
```

Listing 19a

Autor oba programa je profesor Milan Čaberkapa. Prvi program pretvara tastaturu računara u klavijaturu, a drugi ilustruje korišćenje DATA linija za pamćenje nota. Na sličan način možete programirati sviranje bilo koje melodije čije note znate.

Programske datoteke

Unošenje podataka pomoću INPUT naredbe ima više nedostataka. Pre svega, ono bitno usporava odvijanje programa i stvara probleme ako se pogreši pri unošenju podataka. Sem toga, često je potrebno, kao u prethodnom primeru, više puta ponavljati program sa istim ulaznim podacima. Zato se koriste naredbe READ i DATA. Naredba READ radi što i INPUT, samo što umesto sa tastature podatke uzima iz interne datoteke programa koju čine podaci zabeleženi u naredbama DATA. Svakoj promenljivoj navedenoj u listi READ naredbi dodeljuju se redom podaci iz DATA linija. To znači da podaci u DATA linijama obrazuju sekvencijalnu datoteku, odnosno, mogu se čitati samo onim redom kojim su uneti. Informaciju o tome dokle je stigao u čitanju podataka, interpretator ima u posebnoj promenljivoj koja je programeru nedostupna. Programer jedino može da je dovede na početnu vrednost pomoću naredbe RESTORE. Ova verzija bejzika ima mogućnost da se sa RESTORE <n> ukaže da će sledeći podatak biti pročitan s početka DATA liste u liniji <n>. Promenljive u listama READ i DATA naredbi treba da se slažu po vrsti. Nešto više o praktičnoj strani korišćenja READ ... DATA biće reči u objašnjenju upotrebe programa za obradu odeljenjske statistike.

Osmi susret

Računari u školi

Ni kupovina računara za škole ni uvođenje nastave informatike neće postići svoj pravi cilj ako se ne iskorista kao sredstvo za poboljšanje nivoa nastave i olakšice u radu nastavnika. Nivo nastave svakog predmeta može se poboljšati primenom odgovarajućih obrazovnih programa. Najbolje bi bilo da imamo bogato tržište obrazovnog softvera sa koga bismo birali programe prema svojim potrebama. Na žalost, to još dugo nećemo imati, pa moramo i sami ponešto da uradimo.

Na listingu 20 prikazan je program koji može pomoći uvežbavanju prevođenja brojeva iz jednog u drugi brojni sistem. Na sličan način, korišćenjem gotovih modula, koji se prema potrebi povezuju u celinu, možemo brzo sastaviti jednostavne programe koji pomažu u savlađivanju svake nastavne jedinice. Bilo bi dobro da programi maksimalno koriste grafičke mogućnosti računara, jer slika daje znatno više informacije nego tekst.

Odeljenjska statistika

Programa za obradu odeljenjske statistike dat je na listingu 21. Iz komentara uglavnom možete videti šta i kako radi. Ukoliko odlučite da ga koristite, treba da memorišete verziju programa prikazanu na listingu 22 iz koje su izbačeni komentari. Program memorišite bez konkretnih DATA linija da biste mogli da ga koristite za razna odeljenja. Za svako odeljenje treba napisati segment DATA linija u kome se nalaze informacije o uspehu i izostancima i upamtiti u aski formatu da bi se mogao povezati sa programom za obradu.

Na listingu 23 prikazana je struktura DATA linije u kome se nalaze podaci o uspehu nekog odeljenja. Kako program za obradu u kondenzovanoj verziji zauzima samo linije do 900, to pri pisanju datoteke odeljenja možete postupiti na sledeći način.

Prvo sa AUTO 1000,5 zadajte automatsku numeraciju programskih redova. Na početku sva-

```
2
9  *** MENI***
10 CLS  PRINT TAB(33)  BROJN  SISTEMI
PRINT
20 PRINT "1  PROVERA POZNAVA JA BINARN
 OKTALNI I HEKSADEKADNE ..... PRI
NT
30 PRINT "2  PREVODJENJE CELIH BROJ
Z JEDNOG I DRUGI POZICIONI SISTEM  PRINT
40 PRINT "3  PREVODJENJE DI...   I BINA
RNE BROJEVE I P I I
50 PRINT "4  KRAJ RADA"
60 THAT  OTKUCAJTE BR.....
TI',150,20 0 3
70 I$=INKEY$
80 IF I$<>"1" AND I$<>"2" AND I$<>"3" AN
D I$<>"4" ..... 70
90 ON VAL(I$) ..... 100 ..... 00 2 1
```

```
100 C S  PRINT  RACUNANJ  U BROJNIM SIST
E..  OSNOV  Z ...
110 PRINT  1  RACUNANJE .. BINAR O.. SI
ME  PRINT
120 PRINT "2  RACUNANJE  SISTEM  OSNOV
 4   PRI T
130 PRINT  3  ...
... PR  I
140 PRINT "4  RAC.....
SISTEM   PRINT
150 PRINT  ...
160 ...
STI  150,20 0 3
170 I$=INKEY$
180 IF I$<>"1"  AND ... AND I$ > 3
 D I <> 4  ... $ > ... THEN 170
190 ... I$=5  THEN PRIN
200 B=2 VAL(I$)
```

```
210 CLS  PRINT 'RACUNANJE U SISTEMU OSNO
VE  B
220 X=INT(RND*500+1)  Y=INT(RND*500+1)
O=INT(RND*3+1)
230 ON O GOTO 240 250 260
240 O$="+"  Z=X+Y  GOTO 270
250 O$="-"  Z=X-Y  GOTO 270
260 O$="*"  Z=X*Y
270 N=X  GOSUB 700
280 X$=N$
290 N=Y  GOSUB 700
300 Y$=N$
310 N=ABS(Z) GOSUB 700
320 Z$=N$  IF Z<0 THEN Z$="-"+Z$
330 I$="KOLIKO JE "+X$+O$+Y$
340 PRINT P$
350 INPUT S$
360 IF S$=Z$ THEN PRINT "TACNO  BRAVO PRI
```


kog reda otkucajte službenu reč DATA a onda upisujte odgovarajuće podatke koji treba da budu razdvojeni zarezima. U prvoj liniji unesite razred i odeljenje (na listingu 23 to je 4/1). U drugoj DATA liniji unesite broj učenika i broj predmeta. Bitno je da podaci budu uneti ovim redosledom. U svakoj sledećoj DATA naredbi koja se nalazi u parnom broju reda treba upisati prezimena i imena učenika onim redom kojim su upisani u dnevnik. Obratite pažnju da je numeracija DATA linija napravljena tako da srednje dve



cifre broja programskog reda odgovaraju broju u dnevniku odgovarajućeg učenika. Odmah za imenom učenika u narednoj DATA naredbi koja se nalazi u liniji sa neparnim brojem treba uneti redom sve ocene učenika, zatim ocenu iz vladanja, broj opravdanih i broj neopravdanih izostanaka. Znači svaka linija sa ocenama treba da ima tačno P+3 podataka ako je P broj predmeta. Ako je učenik iz nekog predmeta neocenjen odgovarajući podatak treba da bude broj 0.

Kada se upišu sva imena i ocene učenika treba u poslednjoj DATA liniji upisati nazive predmeta onim redom kojim su upisani u dnevniku.

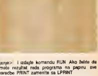
Pre no što iskoristite ove podatke trebalo bi da ih još jednom uporedite sa podacima u dnevniku. Najbolje bi bilo da ih odštampate ako imate štampač. Pošto su ovi podaci deo programa, to postižete vrlo jednostavno otkucavši LLIST. Podatke upamtite sa SAVE "<odeljenja>", A. Dodatak A u SAVE je obavezan.

Listing 21

```
1 '---------------------------------------------------------------
2 '                  PROGRAM ODELJENJSKA STATISTIKA
3 '---------------------------------------------------------------
4 '
10 '-------------------------------
20 'REZERVISANJE MEMORIJSKOG PROSTORA
30 '-------------------------------
40 READ OS$,U P
50 DIM I$(U) P$(P+1),U(U) PR(P+1)
60 DIM O%(U P+3) MP%(P+1 5) M%(U) N%(U)
70 '---------------------------------------------------------------
80 '                  CITANJE PODATAKA
90  I$(I) JE PREZIME I IME I TOG UCENIKA
100  O%(I J) JE OCENA I TOG UCENIKA IZ J TOG PREDMETA
110 'O%(I,J)=0  AKO UCENIK NIJE OCENJEN IZ ODGOVARAJUCEG PREDMETA
120  M%(I) JE BROJ PREDMETA IZ KOJIH I TI UCENIK NIJE OCENJEN
130  N%(I) JE BROJ PREDMETA IZ KOJIH I TI UCENIK IMA SLABE OCENE
140 '---------------------------------------------------------------
150 FOR I=1 TO U  READ I$(I)
160   FOR J=1 TO P
170     READ O%(I J)
175     IF O%(I J)<0 OR O%(I J)>5 THEN PRINT "GRESKA U PODATKU O%(",I ',"' J,")'
ND
180     IF O%(I J)=0 THEN M%(I)=M%(I)+1
190     IF O%(I J)=1 THEN N%(I)=N%(I)+1
200   NEXT J
205   READ O%(I P+1)
210   FOR J=P+2 TO P+3
220     READ O%(I J)
230   NEXT J
240 NEXT I
250 '-----------
260 '  CITANJE NAZIVA PREDMETA
270 '-----------------------
280 FOR I=1 TO P
290   READ P$(I)
300 NEXT I
   '------------------
    RACUNANJE PROSECNE OCENE UCENIKA
   '------------------
      ... 520
   '---
      ISPISIVANJE USPEHA UCENIKA
   '---
      ... 530
   '------------------
      ... PROSECNIH OCENA PO PREDMETIMA
   '------------------
      ... 
   '------------------
      ... PO PREDMETIMA
      ... KRAJU DNEVNIKA
   '------------------
   '------------------
      RACUNANJE PROSECNE OCENE ODELJENJA
   '------------------
      ... 40 (...)
      ... 10 590
   '------------------
      ... ISPISIVANJE USPEHA ... 
   '------------------
      ... NIKA ... OS ... ODELJENJA
6.. ... 45)
650 I?
```

Listing 20

Kada želite da koristite program učitejte ga povežite sa podacima pomoću MERGE <ode-



ljenje> i izdajte komandu RUN Ako želite da imate rezultat rada programa na papiru sve naredbe PRINT zamenite sa LPRINT

... sa nekoliko komentara

Ne slici 24 možete videti rezultate rada programa Program prvo štampa tabelu uspeha učenika. Pri tome eko učenik nije ocenjen iz nekog predmeta umesto uspeha štampe poruku iz koliko je predmeta neocenjen U našem primeru učenik Meus Mikl nije ocenjen iz dve predmeta. Ukoliko učenik ima nedovoljne ocene njegov uspeh se štampa sa porukom koliko ima slabih ocena U našem primeru učenica Bardo Brižit ima jednu a Monro Merlin dve slabe ocene Uspeh se za pozitivno ocenjene učenike računa na osnovu prosečne ocene, ako je vladanje bar vrlo dobro Uticaj ocene iz vladanja može se videti kod učenika Petak Peje i Petak Pete koji bi prema prosečnim ocenama imali bolji uspeh Da se ne bi narušio format štampe znaci posle 18

```
10 REM [illegible]
20 DIM [illegible]
30 DIM O[illegible]
40 FOR I=1 [illegible]
   FOR [illegible]
      READ [illegible]
      I[illegible]
   INPUT[illegible]
      I[illegible]
      I[illegible]
   [illegible]
   [illegible]
   [illegible]
[illegible]
END
[illegible] FOR I [illegible]
[illegible]   FOR [illegible]
[illegible]     [illegible](I)=[illegible](I)+[illegible]
[illegible]   NEXT J
[illegible]   IF N(I)<>0 THEN GOTO 290
[illegible]   U(I)=U(I)/P
[illegible] NEXT I
[illegible] RETURN
[illegible] PRINT TAB(2?),"USPEH UCENIKA [illegible] ODELJ[illegible]"
[illegible] PRINT TAB(26)STRING$(27,45)
[illegible] PRINT
[illegible] PRINT TAB(7),"PREZIME I IME",TAB(35),"USPEH",T
AB([illegible]) "VLADANJE",TAB(66),"IZOSTANCI (O/N)"
[illegible] FOR I=1 TO U
[illegible] PRINT I,"  ",LEFT$(I$(I),18),TAB(26),
[illegible]   IF N%(I)<>0 THEN PRINT "NEOCENJEN IZ",N%(I),"
PREDMETA", GOTO 490
380 IF N"(I)<>0 THEN PRINT "NEDOVOLJAN IZ",N%(I),"
PREDMETA", GOTO 490
390 IF O%(I,P+1)>=4 GOTO 430
400 IF O%(I,P+1)=3 AND U(I)>=4.5 THEN 460
410 IF O%(I,P+1)>=3.5 THEN 450
420 IF O%(I,P+1)=1 AND U(I)>=2.5 THEN 440
430 ON CINT(U(I))-1 GOTO 440,450,460,470
440 PRINT "DOVOLJAN", GOTO 480
[illegible] PRINT "DOBAR", GOTO 480
[illegible] PRINT "VRLODOBAR" GOTO 480
[illegible] PRINT "ODLICAN",
[illegible] PRINT "(",CINT(100*U(I))/100,")",
[illegible] PRINT TAB(54)
[illegible] ON O%(I,P+1) GOTO 510,520,530,540,550
[illegible] PRINT "NEZADOVOLJAVA" GOTO 560
[illegible] PRINT "ZADOVOLJAVA" GOTO 560
[illegible] PRINT "DOBRO", GOTO 560
[illegible] PRINT "VRLODOBRO" GOTO 560
[illegible] PRINT "PRIMERNO"
[illegible] PRINT TAB(68),O%(I,P+2),TAB(74),O%(I,P+3)
[illegible] NEXT I
[illegible] PRINT STRING$(80,45) PRINT
[illegible] RETURN
[illegible] FOR I=1 TO P
[illegible]   FOR J=1 TO U
[illegible]     FOR K=0 TO 5
[illegible]       IF O%(J,I)=K THEN MP%(I,K)=MP%(I,K)+1
[illegible]     NEXT K
650   NEXT J
[illegible]   FOR K=0 TO 5
670   MP%(P+1,K)=MP%(P+1,K)+MP%(I,K)
[illegible]   NEXT K
[illegible] NEXT I
[illegible] FOR I=1 TO P
[illegible]   FOR J=1 TO 5
[illegible]     PR(I)=PR(I)+J*MP%(I,J)
[illegible]   NEXT J
[illegible]   PR(I)=PR(I)/(U-MP%(I,0))
[illegible]   S=S+PR(I)
[illegible] NEXT I
[illegible] PR(P+1)=S/P
780 RETURN
790 PRINT TAB(6)"PREDMET"TAB(20)" ODL   VD   DOB
DOV  POZ  NED  NEO  UKUPNO  P O"
900 PRINT STRING$(70,45)
910 P$(P+1)="UKUPNO"
920 FOR I=1 TO P+1
930 PRINT LEFT$(P$(I),20),
[illegible] PRINT TAB(21)MP%(I,5)TAB(26)MP%(I,4)TAB(31)MP
%(I,3)TAB(36)MP%(I,2),TAB(41)
[illegible] IF I<>P+1 THEN PRINT U-MP%(I,0)-MP%(I,1), ELSE
PRINT I*P-MP%(P+1,0)-MP%(P+1,1),
[illegible] PRINT TAB(46)MP%(I,1)TAB(51)MP%(I,0)TAB(56),
[illegible] IF I<>P+1 THEN PRINT U, ELSE PRINT P*U
[illegible] PRINT TAB(64)CINT(PR(I)*100)/100
[illegible] NEXT I
[illegible] RETURN
```

Listing 22

```
1000 DATA 4/I
1005 DATA 9 I4
1010 DATA ANASTASIJEVIC ATANASIJE
1015 DATA 5,5,5,4,4,4,3 3 5,5,5 4,5 5,5 17,3
1020 DATA BARDO BRIZIT
1025 DATA 4 5 5,4 4,4 I 5,4,5 5 5 5,5 5 19 1
1030 DATA VALENTINO RUDOLF
1035 DATA 5,5 5 4 5 5 3 4,5,5 4,5 5 5 5 9 0
1040 DATA MAUS MIKI
1045 DATA 0 0 I I 3 2 4 5,5 2 3 4 5 3 3 99,20
1050 DATA MAUS MINI
1055 DATA 5 5 5 5,5 5 5,5 5 5,5 5 5 5,5,0,0
1060 DATA MONRO MERLIN
1065 DATA 2 2,2,3,3,I 1 5 5,5 5,5 5,5 3,12,0
1070 DATA PATAK PAJA
1075 DATA 4 5 2 3,4,3,2,5,5 3,2 4 3 2 2,20,25
1080 DATA PATAK PATA
1085 DATA 5,5 5,5 5,5,5,5,5,5,5 5 5 5 3 0,25
1090 DATA HADZIKOSTIC AKSENTIJE
1095 DATA 5 4,5 4 5 4,5 4,5,4,5 4 5 4,5 0 0
1100 DATA SRPSKI  FIZICKO, MARKSIZAM,ANALIZA
ALGEBRA,NACRTNA  ASTRONOMIJA,FIZIKA,BIOLOGIJA
 ENGLESKI  RACUNARI JEZICI SISTEMI KIBERN
ETIKA
```

Listing 23

izbacuju se iz podatka prezime i ime. To je slučaj kod 1. i 9. učenika iz našeg primera.

Posle ispisa uspeha učenika koji možete prekidati sa <BRK> ukoliko informacija pratite na ekranu (i zatim nastavljati ispis sa CONT), sledi ispis uspeha po predmetima. Ispis je uređen kao tabela na kraju dnevnika i zato se imena predmeta u programu skraćuju na zadatu dužinu. Na kraju tabele ispisuje se ukupna statistika odeljenja sa prosečnom ocenom zaokruženom na dve decimale.

Za unete podatke program daje kompletne tabele u zavisnosti od broja učenika i predmeta za oko minut-dva. Koncepcija korišćenja potprogram sprovedena u programa za obradu odeljenjske statistika omogućava da lako dodate nove mogućnosti kao računanje procenata prebrojavanje i sortiranje učenika po uspehu i slično.

USPEH UCENIKA 4/I ODELJENJA

	PREZIME I IME	USPEH	VLADANJE	IZOSTANCI (O/N)	
1	ANASTASIJEVIC ATAN	VRLODOBAR(4.43)	PRIMERNO	17	3
2	BARDO BRIZIT	NEDOVOLJAN IZ 1 PREDMETA	PRIMERNO	19	1
3	VALENTINO RUDOLF	ODLICAN(4 64)	PRIMERNO	9	0
4	MAUS MIKI	NEOCENJEN IZ 2 PREDMETA	DOBRO	100	20
5	MAUS MINI	ODLICAN(5)	PRIMERNO	0	0
6	MONRO MERLIN	NEDOVOLJAN IZ 2 PREDMETA	DOBRO	12	0
7	PATAK PAJA	DOBAR(3 36)	ZADOVOLJAVA	20	25
8	PATAK PATA	VRLODOBAR(5)	DOBRO	0	25
9	HADZIKOSTIC AKSENT	ODLICAN(4 5)	PRIMERNO	0	0

PREDMET	ODL	VD	DOB	DOV	POZ	NED	NEO	UKUPNO	P O
SRPSKI	5	2	0	1	8	0	1	9	4 38
FIZICKO	6	1	0	1	8	0	1	9	4 5
MARKSIZAM	6	0	0	2	8	1	0	9	3.89
ANALIZA	2	4	2	0	8	1	0	9	3.67
ALGEBRA	4	3	2	0	9	0	0	9	4 22
NACRTNA	3	3	1	1	8	1	0	9	3 67
ASTRONOMIJA	3	1	2	1	7	2	0	9	3 22
FIZIKA	6	2	1	0	9	0	0	9	4 56
BIOLOGIJA	8	1	0	0	9	0	0	9	4 89
ENGLESKI	6	1	1	1	9	0	0	9	4 33
RACUNARI	6	1	1	1	9	0	0	9	4 33
JEZICI	5	4	0	0	9	0	0	9	4 56
SISTEMI	8	0	1	0	9	0	0	9	4 78
KIBERNETIKA	6	1	1	1	9	0	0	9	4 33
UKUPNO	74	24	12	9	119	5	2	126	4 24

Slika 24

Samogradnje

Školski računar „tim 011"
Redakcija „Računara" i Institut „Mihajlo Pupin"

Na predlog čitaoca Vlade Badanjaka iz Nove Gradiške

4 TIMSKI DO „tima"

Nenad Dunjić

Niko ne spori da se znanje o računarima najbrže i najlakše stiče uz pomoć samog računara — nikakav metod „štapa i kanapa", odnosno „table i krede" ne može da zameni napoaredno kuckanje po tastaturi. Uvozni računari su, na žalost, i dalje prilično skupi, a njihova je nabavka podložna raznim carinskim komplikacijama, dok su cene domaćih računara u prodavnicama jednostavno astronomske. Zato vam časopis „Računari" i Institut „Mihajlo Pupin" predlažu da sami sagradite računar „tim 011", koji svojim osobinama i cenom može da se meri sa najboljim osmobitnim računarima na svetu.

„Tim 011" je zvanično školski računar u Beogradu i Crnoj Gori koji prema trenutnim kretanjima na uvek naizvesnom obrazovnom tržištu ima velike šanse da bude ozvaničen i u drugim školskim sredinama. Detaljan test „tim 011" uz sagledavanje svih njegovih vrline i nedostataka objavili smo u „Računarima 35", a iscrpan prikaz bejzika objavljujemo u ovom. Ovaj tekst smo pripremili za hardveraše — mladiće i devojke (zašto da ne?) koji su odlučili da sudbinu svog računara uzmu u sopstvene ruke.

Personalni računar „tim 011" razvijen je sa idejom da se postignu što bolje osobine sistema za što manje para. Kako je to uopšte moguće ostvariti?

Postoje dva načina.

Prvi je razvoj sopstvenih visokointegrisanih kola i visokoserijska proizvodnja. Ovaj način kod nas nije moguć. Međutim, činjenica da se poznati svetski proizvođači računara uglavnom ne bave i proizvodnjom integrisanih kola pobija tvrdnje nekih naših stručnjaka koji smatraju da računare ne treba proizvoditi, već ih isključivo treba uvoziti. Izgleda da se ne shvata da se ono znanja koje se razvija i ostaje sa domaćim računarom ne može uvesti.

„Tim 011" . . .

Drugi način je da se malo mučne glavom zasuču rukavi, pažljivo odaberu savremene inostrane mikroprocesorske komponente i dovoljno kvalitetan domači materijal, a zatim štedljivim dizajnom projektuje i proizvede računarski sistem za što povoljnijim odnosom performanse/cena.

Kada se govori o peformansama računara obično se u prvi plan ističe veličina procesorska magistrala podataka. Sigurno je da su 16-bitni a pogotovu 32-bitni procesori moćniji od 8-bitnih. Međutim, zabluda je da se isti aplikativni program izvršava brže samo zbog moćnijeg procesora. Ukupne brzine računara i te kako zavisi od arhitekture i brzina njegovih periferijskih sklopova, a još više od brzina operativnog sistema i načina programiranja. Malo složeniji programi zahtevaju interakciju između korisničkih ulazno-izlaznih uređaja, ugrađenih programa za njihovu podršku, operativne memorije i masovne memorije (diskova). Spora komunikacija ili usko grlo između ovih podsistema može da degradira performanse računarskog sistema više nego što brzina samog procesora to može da popravi. Zato nemojte nimalo da se čudite ako ustanovite da isti grafički program napisan u Turbo Pascal-u na „timu 011" radi brže nego na PC-XT kompatibilnom računaru.

Kombinacija *procesor—periferija—operativni sistem* uz pristupačnu cenu najviše utiče da korisnik bude zadovoljan računarskim sistemom. Osmobitni mikroprocesor je još uvek po ceni najefikasniji za primene u personalnim računarima i industrijskim kontrolerima. Ne zaboravite da su integrisana kola za podršku periferije računara, a takođe i memorije, uglavnom osmobitne arhitetkure. Među 8-bitnim mikroprocesorima Z80 je najpopularniji. Glavni problem sa Z80 je adresiranje memorije ograničeno na 64 K, koje praktično onemogućava korišćenje ozbiljnijih prevodilaca i aplikativnih programa dobro poznatih vlasnicima PC računara. Hitačijev HD64180 visokointegrisani CMOS mikroprocesor direktno adresira 512 K memorije i poseduje nadskup Z80 instrukcija čije je izvršavanje ubrzano. Cena računara koji koriste ovaj mikroprocesor drastično je smanjena integracijom više ključnih periferijskih sklopova na samom čipu. Zato je bez dvoumljenja izabran za procesor računara „tim 011". O osobinama Z-operativnog sistema već smo pisali na stranicama „Računara" 34, 35 i 36 i 37, a želja nam je da se kroz akciju samogradnje računara „tim 011" što pre direktno upoznate s njim.

Tim 011 kao telefon



Samogradnji računara „Tim 011" u Institutu „Mihailo Pupin" i Beogradskoj računarskoj industriji (BRI) pristupamo s velikim entuzijazmom i uverenjem da doprinosimo opštoj društvenoj informatičkoj pismenosti. Odrekli smo se zarade i pokrivamo samo troškove proizvodnje, ubeđeni da će veliki broj ljudi, pre svega mladi zaljubljenici u računarstvo biti u stanju da improvizuju mnoge podsklopove.

Samogradnju treba posmatrati fleksibilno, tako da u njoj mogu da učestvuju i oni koji će nabaviti samo najkritičnije delove i oni koji će sastavljati gotove podsklopove. Akciju ćemo smatrati uspešnom ukoliko računar „tim 011" uđe u mnoge domove, škole i organizacije, ukoliko se o njemu priča i piše među mladima, ukoliko se za njega prave programi i društva korisnika koja će se takmičiti u inovacijama — drugim rečima, ukoliko računar „tim 011" postane ono što je danas telefon ili televizor.

Dr Dragoljub Milićević
direktor RJ računarstvo Instituta „Mihailo Pupin" i predsednik KO BRI-ja

Tehničke karakteristike

„Tim 011" čine:

— procesor HD64180, takt 6 MHz
— 256 K operativna memorija
— 8 K ROM sa programima za testiranje računara i učitavanje operativnog sistema
— grafički kontrolor rezolucije 512×256 tačaka u 4 boje / intenziteta sa 32 K grafička memorija
— disk kontroler za četiri 5 1/4 ili 3 1/2 inča diskatne jedinice maksimalnog kapaciteta 800 K
— dva serijska RS232 interfejsa
— paralelni CENTRONICS interfejs
— ulazno-izlazna magistrala za proširenje računara.

. . . i njegov procesor

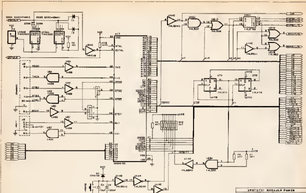
Svoje najbolje hardverske osobine „tim 011" ima zahvaljujući uglavom mikroprocesoru HD64180.

Na blok shemi (sl. 2) prikazano je 5 funkcionalnih celina mikroprocesora HD64180.

CPU

Centralni procesor je mikrokodiran tako da izvršava nadskup Z80 instrukcija. Sve Z80 instrukcija se na HD64180 izvršavaju za manje takt ciklusa, što znatno ubrzava rad postojećih programa pisanih za mikroprocesor Z80. Skup instrukcija HD64180 uključuje i 10 potpuno novih instrukcija. Mnemonička imena ovih instrukcija su SLP, MLT, INO, OUTO, OTIM, OTIMR, OTDM, OTDMR, TSTIO i TST.

SLP instrukcija postavlja mikroprocesor u „sleep" (spavaj) režim rada sa smanjenom potrošnjom struje. Ona se ne koristi na „tim 011", ali je korisnici mogu upotrebiti u svojim programima.



Tim 011 — mikroprocesor i oscilator

MLT instrukcija množi dva bajta za 17 takt ciklusa. Ona znatno ubrzava računanja u programima koji je koriste

Preostala instrukcija izvode ulazno-izlazne operacije sa procesorskim registrima prenos bloka podataka iz memorija na ulazno-izlazni port i nedestruktivne i logička operacije sa registrima portovima i podacima

SLOŽENI TAKT GENERATOR (TIMING GENERATOR)

Iz osnovnog takta kristalnog oscilatora generiše složeni takt za interne sklopove mikroprocesora.

KONTROLER MAGISTRALE (BUS-STATE CONTROLLER)

Kontroliše procesorsku magistralu opslužuje stanja čekanja memorija i ulaza izlaza izvršava RESET, osvežava sadržaj dinamička memorije i upravlja ustupanjem magistrala pri direktnim memorijskim (DMA) prenosima podataka.

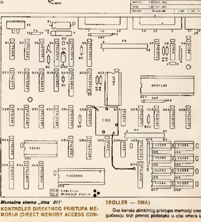
KONTROLER PREKIDA (INTERPUT CONTROLER — INTC)

Nadzire i daje prioritet hardverskim zahtevima za prekid iz 8 internih i 4 spoljna izvora prekida. Više načina odgovora na prekid zadaje se programski

SKLOP ZA UPRAVLJANJE MEMORIJOM (MEMORY MANAGMENT UNIT — MMU)

Ovaj sklop dodeljuje logičkom adresnom prostoru procesora od 64 K fizički adresni prostor od 512 K, koristeći efikasan metod zajedničkih područja i memorijskih banki

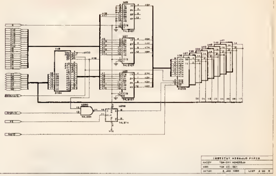
Preostale četiri funkcionalne celine mikroprocesora HD64180 su ugrađene ulazno-izlazne periferija



Montažna shema „tima 011“

KONTROLER DIREKTNOG PRISTUPA MEMORIJI (DIRECT MEMORY ACCESS CONTROLLER — DMA)

Dve kanala direktnog pristupa memoriji omogućavaju brzi prenos podataka u oba smera iz:



„Tim 011“ memorija

memorija u memoriju memorije u memorijski ulaz-izlaz i memorije u ulazno-izlazni port DMAC adresira svih 512 K fizičke memorije Pri učestanosti takta od 6 MHz brzina pranosa iznosi 1 Mb/s

ASINHRONI SERIJSKI KOMUNIKACIONI INTERFEJS (ASCI)

Sastoji se od dva dvosmerna (full-duplex) univerzalne asinhrone ulaze-izlaza i programabilnog generatora bodne učestanosti kojim se bira brzine pranose ASCI može da koristi DMAC za brzi serijski pranos podataka što znatno smanjuje angažovanje centralnog procesora oslobađajući ga za važnije poslove

SINHRONI SERIJSKI ULAZ-IZLAZ (CSI/0)

Sastoji se iz polu-dupleksnog serijskog ulaza-izlaza Može da se koristi za veoma brze serijske prenose podataka Na primer između dva mikroprocesora u višeprocesorskom računeru Nije iskorišćen u timu 011 (što ne znači da tako i treba da ostane)

PROGRAMABILNI ČASOVNIK-BROJAČ (CTC)

Sastoji se od dve nezavisna 16-bitna programabilna brojača i njihovih registara Takt za brojače dobijen je deljenjem osnovnog takta (6 MHz) sa 20 Tim 011 ih koristi za časovnike realnog vremena ili kao generatore prekida u multiprogramskim aplikacijama

Posle ove veoma šture priče o procesoru HD64180 (detaljen opis objavljujemo u aprilskom broju Računara) verovatno vam je jasnija zašto jedan relativno složen računar kao što je tim 011 ima samo četrdesetak integrisanih kola

Naš sledeći zadatak je da pokušamo da vam objasnimo hardver računara tim 011 Shema kompletnog računara podeljena je na 5 delova koji su zasebne funkcionalne celine Svi signali su označeni mnemoničkim imenima i uokvireni u kućice koje su različite za ulazne izlazne i biderakcione signale Imena se sufiksom/0 označavaju signale čije je aktivno stanje nula Integrisana kola su označena brojevima a slovo na kraju oznaka upotrebljeno je za označavanje određenog sklopa kod kola koja se sastoje iz više istovetnih sklopova Izvodi za napajenje kola i kondenzatori za njihovu blokadu nisu označeni na shemi

Centralni procesor

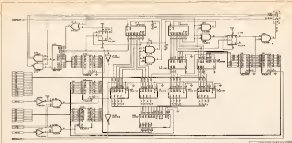
Na shemi Tim 011 CPU prikazani su HD64180 mikroprocesor

Generator takta, koji formiraju kvarcni oscilator (O-OCS) i brojači U23AA i U23AB (74HC4040) proizvodi takt za disk kontroler i zajednički takt procesora i grafičkog kontrolera. Ostavljena je mogućnost ugradnje dva tipa disk kontrolera FDC 9266 i FDC 9268 Oba kontrolera su funkcionalno identična ali im se razlikuje takt učestanost Ako se upotrebi FDC 9266 prekida se veza prema nožici 10 kola U23A i ugrađuje kvarcni oscilator od 24 MHz. U ovom slučaju takt za disk kontroler je 8 MHz (dobijen deljenjem 24 MHz sa 3 brojačem U23AA) a takt za procesor i grafički kontroler 12 MHz (dobijen deljenjem sa 2 brojačem U23AA) Za FDC 9268 upotrebljava se kvarcni oscilator od 48 MHz i prekida veza se nožicom 11 kola U23A Takt za disk kontroler je 16 MHz (48/3) a za procesor i grafiku 12 MHz (48/4) Procesorski takt se u samom mikroprocesoru deli sa 2 i dobija se osnovni takt (signal FI) učestanosti 6 MHz.

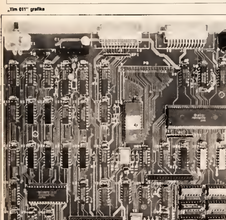
Sklop za generisanje stanja čekanja (u trenutku kada procesor prihvata instrukciju) neophodan je zbog nedovoljne brzine dinamička memorije Ako nabavite memorije sa vremenom pristupa od 100 ns ne ugrađujte kolo U7

PCW brzinski testovi

Računar	Jazik	Pros	intma	realma	triog	txtscr	grfscr	store
BBC Arhimed	Basic 5	2.99	0.21	0.25	1.00	3.36	6.53	6.58
Compaq 386	GWbasic	7.61	1.00	0.96	3.85	25.50	4.80	9.60
IBM PS/2 50	IBM Bas	11.74	1.45	2.04	12.50	27.90	7.93	10.70
Tendom PAC286	Basica	14.70	2.00	2.00	15.00	47.00	12.00	10.20
IBM PC AT (6 MHz)	Basica	14.97	1.01	1.89	4.17	25.35	46.50	10.92
IBM PS/2 30	Basica	15.91	2.60	3.40	25.40	36.30	14.20	13.60
BBC B+65C02	Basic 2	16.58	1.92	3.95	53.30	6.55	10.85	22.90
Meeter Compact	Basic 4	20.17	2.22	4.62	33.20	19.40	22.40	39.20
TIM 011	GBASIC	22.05	5.40	5.50	33.80	37.00	13.10	37.50
Standardan BBC B	Basic 2	24.67	2.60	5.70	60.50	13.70	21.20	24.30
Atari 520 ST	FBasic	28.79	0.62	0.84	3.20	120.80	17.90	29.40
IBM PC (4.77 MHz)	Basica	37.93	6.20	8.20	47.00	100.00	49.00	17.20
Amstrad 6128	Basic	39.76	4.50	7.60	16.30	159.60	22.00	28.60
Sinclair QL	Basic	39.77	7.70	6.40	27.70	28.60	149.40	18.80
Amiga 2000	Basic	52.16	3.19	4.35	19.25	137.16	116.46	32.50
ZX Spectrum	Basic	91.50	—	17.50	226.8	84.10	83.50	45.80



„Tim 011" grafika



TIM 011 — elektronske komponente

Otporne dekade

4K7	2 kom.
330	1 kom.

Otpornici:

100	1 kom.
330	3 kom.
470	2 kom.
1K	4 kom.
4K7	5 kom.

Elektrolitski kondenzatori:

22 F 40V	1 kom.
10 F/16V	1 kom.

Keramički kondenzatori:

27pF	2 kom.
330pF	1 kom.
1nF	1 kom.
1.5nF	1 kom.
100nF	48 kom.

Kristali

12.288MHz	1 kom.*
8 MHz	1 kom.

Taster za reset

Konektori

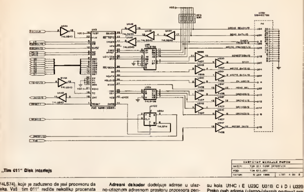
DIN 5-polna utičnica
IDS34 ili IDC34B 2 komada
DB25P
DB25S

U verziji za samogradnju kristali se zamenjuju oscilatorskim sklopom koji obuhvata sledeće delove:
SCC 25 000 integrisani CMOS kristalni oscilator
U23 74HC393
1N4148 dioda 2 komada i jedan otpornik 4K7

Poluprovodnici:

U1	74S08
U2	74LS08
U3	U13 HC4040
U4	U5, U6, U40, U41 74LS374
U7	U11 U15 U25 74LS74
U8	MC1489
U9	MC1488
U11	U12 74LS283
U14	U44 74LS240
U18	74LS20
U19	U20, U21 U22 U26 U27 U28/74LS157
U23	74LS00
U24	74S04
U29	74S153
U30	43256 statički RAM
U16	2764 EPROM
U17	HD64180 mikroprocesor
U31	74LS32
U32	74LS138
U33	7407
U34	7402
U35	74LS158
U36	U37, U38 U39 U45, U46, U47 U48, 41256 DRAM
U42	74LS244
U43	FDC9266 flop i kontroler

Štampano kolo sa montiranim komponentama



„Tim 011" Disk interfejs

(74LS74), koje je zaduženo da javi procesoru da čeka. Vaš „tim 011" radiće nekoliko procenata brže.

> **Satnica akcije**
>
> *Da bi samogradnja uopšte imala smisla, moramo da obezbedimo štampano kolo, programiranje EPROM-a i sistemski softver na disketi. To će ujedno biti minimalni kit. Institut „Mihajlo Pupin" će, u saradnji sa svojim kooperantima, zainteresovanima obezbediti izvore za napajanje i kutije koje će se unekoliko razlikovati od komercijalne verzije predstavljene u „Računarima 35" — kompletna elektronika je smeštena u kutiju sa monitorom što znači da se računar sastoji samo od monitora i tastature i da zauzima vrlo malo mesta na radnom stolu. Što se tastature tiče, moći ćete da se opredelite između standardne Pupinove „TIM" tastature i bilo koje PC tastature koju nabavite u zemlji ili inostranstvu — malom modifikacijom projekta za potrebe samograditelja komunikacija sa tastaturom je prilagođena PC standardima.*
>
> *U ovom specijalnom izdanju objavljujemo preliminarnu narudžbenicu čije vas slanje ni na šta ne obavezuje — želeli bismo da što pre procenimo potencijalni broj zainteresovanih za samogradnju kako bismo bili u prilici da precizno ugovorimo sve uslove sa proizvođačima komponenti. Orijentaciona cena štampanog kola mogla bi da se kreće između 30 000 i 50 000, programiranje EPROM-a i sistemskog softvera bilo bi praktično besplatno (plaćaće se samo režijski troškovi), dok će „tim" kutija zajedno sa monitorom i izvorom za napajanje verovatno koštati ispod 30 starih miliona. Ako zanemarimo pasivne komponente koje se kod nas lako nabavljaju, ostaje vam da rešite problem nabavke integrisanih kola koja naš časopis, jasno, ne može da kupuje u inostranstvu. Zato ćemo napraviti dogovor sa nekim renomiranim inostranim distributerom komponenti koji će, poštujući naše poštanske propise, zainteresovanima slati kompletan kit koji će obuhvatiti mikroprocesor, disk kontroler, RAM, EPROM, sva TTL kola i samu disk jedinicu od 3.5 inča. Komplet ćete moći da naručite posredstvom banke ili, ukoliko ste vlasnik neke od deviznih kreditnih kartica, telefonom. Zar ne zvuči jednostavno?*
>
> *Što se oživljevanja ploča tiče, „tim 011" je računar relativno pogodan za samogradnju, što znači da će veći deo računara svakako proraditi „iz prve". Treba naravno voditi računa i o onima kojima je pre uključivanja kompjutera crna mačka prešla put, u saradnji za kolegama koji su oživljavali ploče za TIM-ove iz prve Pupinove serije organizovaćemo neku vrstu servisa koji će besplatno ili uz minimalne cene pomagati pri ovom osetljivom poslu.*
>
> *Za razliku od samogradnje „galaksije", ova će akcija biti pod punom kontrolom redakcije „Računara" — narudžbenice će se kao i ranije slati na našu adresu ali ćemo ovoga puta mi kompletirati i slati čitav kit. Na taj način ćemo u svakom trenutku znati kako stvari stoje i koje probleme treba rešiti pa ćemo biti u prilici da zainteresovane čitaoce pravovremeno obaveštavamo kako posredstvom časopisa tako i telefonom. Tim 011 će tako na najlakši mogući način ući u mnoge domove.*

Adresni dekoder dodeljuje adrese u ulazno-izlaznom adresnom prostoru procesora periferijama računara. Za ovu funkciju upotrebljena su kola U14C i E, U23C, U31B, C i D i U32B. Preko ovih adresa (ulazno-izlaznih portova) procesor komunicira sa grafičkim kontrolerom (signal IOE i SCROLL), disk kontrolerom (signali FDCSEL/0 i FDCDMA/0), paralelnim CETRONICS portom (signal PRINT/0) i eventualnim proširenjima računara (signal EXPSEL/0). Tabela 1. prikazuje ulazno-izlazne adrese računara „tim 011".

Ime	adrese	koristi se
FDCSEL	od 80h do 9Fh	80h i 81h
FDCDMA	od A0h do BFh	A0h
PRINT	od C0h do CFh	C0h i C1h
SCROLL	od D0h do DFh	D0h
EXPSEL	od E0h do FFh	XXh

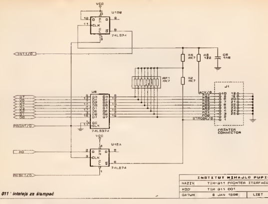
Sklop za RESET sastoji se od invertora U14B, D i H. Po uključenju računara obezbeđuje RESET signale za procesor i disk kontroler.

Memorija

Na shemi „Tim 011 MEMORIJA" prikazani su ROM i RAM računara.

ROM zauzima donjih 256 K adresnog prostora pošto se selektuje ako je adresni vod A18 nula. Kapacitet ROM-a na pločici računara je 8 K, te se njegov sadržaj ponavlja 32 puta u donjih 256 K adresnog prostora. Kome se ovo ne sviđa može relativno jednostavnom intervencijom da proširi kapacitet ROM-a do mogućih 256 K (pod uslovom da ima čime da ispuni toliki ROM).

RAM okupira preostalih gornjih 256 K adresnog prostora. Upotrebljeno je 8 standardnih dinamičkih memorija 41256—15 kapaciteta 256 K·1 bit. RAM interfejs je veoma jednostavan zahvaljujući činjenici da HD64180 sam generiše adrese za osvežavanje sadržaja dinamičke memorija. Za razliku od Z80, HD64180 može da se programira da vrši osvežavanje memorije na



„Tim 011" intefejs za štampač

svakih 10, 20, 40 ili 80 FI (takt) ciklusa. Time se takođe postiže ubrzanje u odnosu na Z80. „Tim 011" osvežava sadržaj dinamička memorija svakih 80 FI ciklusa. Kao adresni multipleksori se koriste U26, U27 i U28 (74LS157). Signal WR/0 mikroprocesora direktno je povezan sa WR nožicom RAM-a. Procesorski signal MEMENA/0 generiše RAS signal (izbor adrese vrste) dinamičke memorije, a CAS (izbor adrese kolone) signal se generiše na vodeću ivicu FI signala, koji sledi vodeću ivicu E signala. Ovo ostavlja dovoljno vremena za adresne multipleksore da se postave pošto vodeća ivica signala E prebacuje multipleksore sa adrese vrste na adresu kolone dinamičke memorije.

Grafika

Svi popularni personalni računari obično koriste specijalno razvijene grafičke čipove. Analizirajući odnos performanse/cene onih čipova koji su namenjeni širokom tržištu (mogu se nabaviti) došli smo do zaključka da je efikasnije razviti sopstveni grafički kontroler (rezolucije 512·256 tačaka) sa korišćenjem običnih TTL čipova.

Osnovni zadatak koji vrši grafički kontroler je paljenje i gašenje tačaka na ekranu monitora računara. Za obavljanje ovog posla potrebna je velike brzina. Na računaru „tim 011" jedna tačka se ispisuje za dvanaestomilioniti deo sekunde. Međutim, obraćanje grafičkoj memoriji nije tako često. Jednim obraćanjem grafičkoj memoriji (U30, 43256) kontroler dobija podatke o četiri susedne tačke. Organizacija grafičke memorije je bajtovska, a u jedan bajt staju tačno četiri tačke opisane sa po 2 bita, tj. 4 boje/intenziteta. Sadržaj četiri susedne tačke upiše se u registar (kolo U40, 74S374) a zatim se brzinom od 12 MHz (učestanost signala DOTCLK) bitovi tačaka odabiraju sa dve selektora koji imaju svaki po četiri ulaza i po jedan izlaz (kolo U29 74S153). Na izlazima selektora se formiraju signali (VID1 i VID2) koji se po ukrštanju sa signalom DOTCLK (kolo U1A i D, 74S08) vode na video stepen monitora, u koma se pretvaraju u signale različite amplitude. Amplituda signala nastaloj od signala VID2 je veća od amplitude signala nastalog od VID1. Prisustvom jednog, drugog ili oba signala dobijamo tri vidljiva intenziteta, a odsustvom oba — crno.

Poslovi oko ispisivanja slika na ekran obavljaju se bez učešća procesora. Njegov zadatak je samo da jednom upiše sadržaj slike u grafičku memoriju, a zatim je slobodan za druga poslove sve dok se ne pojavi potreba da se sadržaj ekrana promeni. Brojači (U13 i U3 74HC4040) neprekidno generišu adresu video memorije sa koje se čita podatak o četiri susedne tačke za svako od 128 mesta (128·4=512) svake od 256 linija na ekranu. Pored ovoga brojači uz pomoć kola (U1B i C, U2A i D i U10B) naizmeno proizvode impulse za horizontalni i vertikalni povratak (gašenje) elektronskog snopa katodne cevi. Kola U31A i U10A. Isključivanjem selektora garantuju odsustvo video signala za vreme povratnih intervala. Po smeštanju četiri tačke u registar, brojač (U13 preko invertora U24B) automatski dodeljuja procesoru posredstvom multipleksora (U19—U22, 74LS157) upravljanje adresama grafičke memorije. Ovo se odvija toliko brzo da procesor nikada ne čeka na pristup grafičkoj memoriji. On je uvek vidi spremnu i na časeći ni časak (posredstvom kola U24F, U18A i U41) može da pročita ili (posredstvom kola U14F, U18B i U42) upiše sadržaj na željenu adresu grafičke memorije.

Kako se procesor snalazi sa adresama grafičke memorije? Doste dobro. On ih vidi kao gornjih 32 K svog ulazno-izlaznog adresnog prostora

---

ZCPR3

Ime TIM-ovog operativnog sistema ZCPR3 nije mnogo poznato ali po pominjanju magične skraćenice CP/M sve postaje jasnije — ZCPR3 je zapravo nagradnja CP/M-a 2.2 prilagođena mikroprocesoru HD-64180. Samo se po sebi razume da je i pored brojnih poboljšanja sačuvena potpuna kompatibilnost sa ogromnom već postojećom bibliotekom CP/M programa, kmja je na primer dobro poznata vlasnicima Amstradovih računara ili Komodora 128. Poboljšanja se svode na rezidentna komande, ugrađeni editor naredbi, komandna procedura, redirekciju i mnogo drugih stvari. Posebno je zanimljivo da je ZCPR3 takozvani public domain operativni sistem što znači da ga možete slobodno umnožavati i distribuirati štiteći jedino moralna prava njegovog autora.

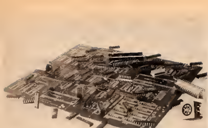
Kako dalje

U ovom broju objavljujemo kompletnu dokumentaciju za samogradnju komercijalne verzije „tima 011" — izuzev štampanog kola. Ono je previše komplikovano za izradu u kućnim uslovima, a cena komercijalne verzije je dovoljno niska da bi se njegova samogradnja isplatila bilo kome.

Da bismo pojednostavili samogradnju „tima" i olakšali nabavku delova, predvideli smo nekoliko modifikacija komercijalne verzije:

- *interfejs za IBM PC kompatibilnu tastaturu*
- *izvor napajanja sa mrežnim transformatorom*
- *jedinstveni oscilatorski sklop sa integrisanim oscilatorom*

Modifikacije će biti opisane u „Računarima" 38 i 39. Iako je već i ovaj prikaz dovoljan iskusnijim samograditeljima da s uspehom sklope svoga „tima", za sledeći broj „Računara" pripremamo detaljna uputstva i praktične savete za sklapanje računara.



(adrese od 8000 h do FFFFh) Ovo je ostvareno signalima IOE i A15 koji se dovode na i kola U18A i B

Na kraju priče o grafičkom kontroleru rasvetlimo ulogu sklopa koji čine U5 U4 U11 i U12 On služi za zadavanje linija od koje počinje ispisivanje slike na ekran Već pogađate da je reč o hardverskom pomeranju slike po vertikali (scroiling) Broj linija se zadaje ulazno-izlaznom portu SCROLL (adresa do D0h do DFh) I time pamti u registru U5 (74LS374) Uloga registra U4 (74LS374) je da pamti podatak iz registra U5 sinhrono sa vertikalnim impulsom (trenutak nastajenja nove slike) Na ovaj način je uklonjeno neprijatno treperenje koje nastaje ako se slika pomeri pre nego što se cela ispiše Kola U11 i U12 sabiraju podatak registra U4 sa vrednošću brojača U3 a rezultat je deo adresa grafičke memorije koji određuje vertikalnu poziciju slike pohranjene u memoriji

Disk

Prvi pogled na shemu „Tim 011" DISK INTERFACE kazuje da se radi o veoma jednostavnom sklopu Jednostavnost je ostvarena upotrebom savremenog disk kontrolera FDC 9266 U čipu se nalazi kontroler kompatibilan sa kontrolerima NEC 765 ili Intel 8272 i digitalni separator disk podataka Takt za 9266 iznosi 8MHz (FDCCLK) Gledajući vezu HD64180 — 9266 sa procesorske strane za poslove inicijalizacije i proveru statusa kontrolera upotrebljen je progamirani ulaz-izlaz, a za prenos podataka DMA

Programirani ulaz-izlaz ostvaren je vezivanjem ulaza CS/0 za selektovanje disk kontrolera sa ulazno-izlaznim vodom FOCSEL/0 (adresa od 80h do 9Fh) i direktnim povezivanjem signala RD/0 i WR/0 koje generiše HD64180 sa odgovarajućim ulazima kontrolera. Izbor statusnog ili podatkovnog registra kontrolera vrši se procesorskim adresnim signalom A0 Napomenimo da je ovakav programirani ulaz-izlaz krajnje uobičajen za sve periferijske čipove Z80 familije

Za DMA je upotrebljen kanal 1 (kontrolni signali DREQ1/0 i TEND1/0) pošto su kontrolne linije kanala 0 multipleksirane sa ASCI takt signalima Iako se ovi takt signali ne koriste u „tim-u 011" ostavljeni su za eventualnu buduću upotrebu

DMA prenos započinje 9266 signalom DMRQ (zahtev za DMA) koji izaziva nulu na nožici procesora DREQ1 Potom HD64180 izvršava DMA čitanje/pisanje na ulazno-izlaznom portu FDCDMA (adresa od A0h do BFh) što izazivaju nulu na DACK ulazu (za potvrdu DMA prenosa) disk kontrolera Po završetku prenosa programom zadatog broja podataka HD64180 izdaje signal TEND1/0 koji posle inverzije (U14A) dolazi do TC ulaza disk kontrolera da označi kraj DMA prenosa Po završenom prenosu 9266 šalje procesoru signal za prekid (vidi na shemi INT U44C INT2/0) koji poziva prekidnu rutinu za proveru statusa 9266 radi detekcije eventualnih grešaka

Flip-flop U25A (74LS74) upotrebljen je da obezbedi kašnjenje početka DMA prenosa 9266 zahteva period latence (mirovanja) u trajanju od 800 ns od pojave DMRQ signala do početka DMA prenosa Pošto je veoma brz HD64180 odgovara na zahtev za DMA transfer pre nego što 9266 može da prihvati podatke Ovaj problem rešen je time što je DMRQ signal zakašnjen za jedan ciklus osvežavanja memorije (signal RFSH/0)

Kolo U32A (74LS139) upotrebljeno je za dekodovanje signala za izbor jedne od četiri disketne jedinice Signal TXS sinhronog serijskog ulaza izlaza upotrebljen je za uključivanje/isključivanje motora disketnih jedinica Preostala kola upotrebljena su kao baferi za disketnu magistralu

Printer

Paralelni printer interfejs se sastoji od 8-bitnog registra U6 (74LS374) i dva flip-flopa U15A i B (74LS74) Izveden je po Centronics standardu i upravlja se ulazno-izlazni portom PRINT (adresa od C0h do CFh)

Slanje podataka štampaču počinje upisom bajta na ulazno-izlazni port C1h Pored podataka štampač dobija i signal koji označava prispeće novog podatka (STROBE/0) Slanje istog bajta na port C0h ukida signal STROBE/0 Kada štampač procesira podatak, on šalje signal ACK/0 koji posredstvom flip-flopa U15B generiše prekid (INT1/0) Rutina za opsluživanje ovog prekida ukida zahtev za prekid slanjem proizvoljnog podatka na port C0h Generisanje prekida obezbeđuje realizaciju aplikacija u kojima se štampanje obavlja paralelno sa drugim poslovima

Izvor napajanja

U komercijalnoj verziji „tima 011" koristi se prekidački izvor napajanja (+5V−2A 12V−1A −12V−50mA) ali on na žalost nije pogodan za samogradnju Upravo je u toku razvoj klasičnog ispravljača a shemu ćemo objaviti u „Računarima 38"

Ako naše nastojanje da vas upoznamo sa hardverom računara „tim 011" nije do kraja urodilo plodom nemojte da se obeshrabrite Ovo je samo početak priče o „timu" Slede nastavci o neophodnim modifikacijama komercijalne sheme (klasičan izvor napajanje i interfejs za PC tastaturu) koja će pojednostaviti samogradnju kao i napisi u kojima će „tim 011" biti osvetljen iz programerskog ugla

Srećno sklapanje!

TIM 011 i PC

Dok smo 1982 našu „galaksiju" poredili sa Sinklerovim ZX-81, „tim 011" možemo bez mnogo straha porediti sa IBM PC-jem vodećim poslovnim računarom osamdesetih godina Bez mnogo razmišljanja mora se reći da IBM PC AT sa megabajtom RAM-a i hard diskom od 40 megabajta pa čak i IBM PC XT sa hard diskom od 20 megabajta predstavljaju značajno moćnije mašine od „tima 011" Treba ipak imati u vidu da najjeftiniji ovako konfigurisan AT klon košta preko 3500 DM plus odgovarajući iznos carine i drugih dažbina dok IBM XT sa hard diskom košta 2000 maraka+carina IBM PC XT bez hard diska bi i dalje koštao dvostruko više od „tima 011" (predviđa se da će „tim 011" u samogradnji koštati 50—60 miliona starih dinara) pri čemu se ne može sa sigurnošću reći da je IBM [illegible] korisnija — PC bez hard diska iskreno govoreći nije naročito upotrebljiv računar dok je TIM 011 sa svojim disketama dvostrukog kapaciteta itekako upotrebljiv za sve poslove PC DOS softverska biblioteka je naime zasnovana na postojanju hard diska dok se kompletna CP/M biblioteka razvila u danima kada je čak i druga disk jedinica predstavljala priličan luksuz!



Elektronska industrija Niš OOUR Fabrika računskih mašina" nudi vam tri modela PC računare IBM kompatibilnih To su PC računari FRS 5100 FRS 7100 i FRS 9100 vrhunske tehnologije i modularna konstrukcija sa širokim izborom dodatne opreme

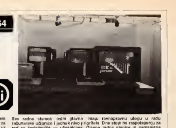
Model FRS 5100 radi kao ekonomična inteligentna radna stanica dok je model FRS 7100 predviđen za rad u oblastima gde se zahteva velika brzina i česta upotreba fiksnog diska za profesionalnu obradu podataka.

U osnovi imaju INTEL-ov 16-bitni procesor 8088-2 sa osnovnim taktom od 4 77/8 MHz Svaki od ovih modela sadrži po 640 KB RAM-a i podnožje za aritmetički koprocesor INTEL 8087-2 kao i po 8 slotova za proširenja HT kompatibilni Kod oba modela jedan slot je

MMS 1800 RU

Ⓔ PECOM 64

RAČUNARSKA UČIONICA



MMS 1800 RU je mikroračunarski sistem namenjen osnovnom i srednjem obrazovanju. U procesu obrazovanja ovakav sistem se može koristiti za obavljanje edukativnih poslova: nastave uz pomoć računara, učenje uz pomoć računara, učenje programskih jezika i za obavljanje drugih poslova u obrazovnim institucijama (INDOK delatnost, upravljanje vaspitno-obrazovnim poslovima i administrativno-statistički poslovi).
Sistem je projektovan na bazi modularnog mikroračunarskog sistema MMS 1800 RU kao glavnog računara, na koji se priključuje maksimalno 16 radnih stanica. Ovakva konfiguracija omogućuje rad više korisnika uz sekvencijalan pristup glavnom računaru i korišćenje svih njegovih resursa.
Radna jedinica je realizovana sa mikroračunarom PECOM 64 i monitorom.
Režimi rada u kojima može da radi ova stanica su:

- — terminalski režim rada i
- — autonomni režim rada.

Radne stanice komuniciraju sa centralnim procesorom preko asinhrone veze (RS 232C) u terminalskom režimu rada. U autonomnom režimu rada gube se terminalske funkcije i PECOM 64 radi kao autonomni mikroračunar.

HARDVER SISTEMA

U osnovnoj konfiguraciji sistem računarske učionice (RU) sadrži:

- — centralnu jedinicu MMS 1800 RU (sa multipleksiranim UART-om za povezivanje radnih stanica — maksimalno 16 PECOM-a 64)
- — radne stanice (maksimalno 16) koje su u stvari μR PECOM 64 sa firmverom za rad u računarskoj učionici
- — operatorski terminal koji može da bude PECOM 64 sa monitorom ili asinhroni video terminal VIP 7251 RU
- — serijski štampač (80—132 kolonski) sa interfejsom RS 232 C

Komunikacioni deo omogućuje multipleksiranu vezu radnih stanica sa centralnim procesorom. Svakom korisniku je omogućeno korišćenje resursa MMS 1800 RU (čitanje, upis, kreiranje, kopiranje datoteka i slično).

SISTEMSKI SOFTVER CENTRALNE JEDINICE

Sistemski softver računarske učionice čine sistemski softver centralne jedinice i sistemski softver mikroračunara PECOM 64.

Sistemski softver centralne jedinice MMS 1800 RU čine:

- — uslužni program MS 10 (firmver) koji obezbeđuje:
  - — komunikaciju MMS 1800 RU sa mehanizmom disk jedinice
  - — interfejs MMS 1800 RU sa operatorskim terminalom
  - — pristup svim memorijskim lokacijama i registrima opšte namene (SPR)
- — centralna jedinica MMS 1800 RU
- — MDOS operativni sistem na disketi
- — program za podršku rada MMS 1800 RU u računarskoj učionici
- — programski jezici (opcijski)

BASIC jezici (opcijski)
PLM 180
μFORTH

SISTEMSKI SOFTVER MIKRORAČUNARA PECOM 64

Sistemski softver mikroračunara PECOM 64 sastoji se iz sledećih elemenata:

- — monitorskih programa za podršku rada mikroračunara PECOM 64 u autonomnom režimu
  - — rad sa tastaturom
  - — upis i čitanje sa trake
  - — programi za rad sa mašinskim jezikom mikroprocesora CDP 1802
- — programskih jezika
  - — BASIC-a
  - — asemblera
  - — editora
- — programa za podršku rada mikroračunara PECOM 64 kao operativnog terminala u računarskoj učionici
- — programa za podršku rada mikroračunara PECOM 64 kao radne stanice u sistemu računarske učionice

Sve radne stanice osim glavne imaju ravnopravnu ulogu u radu računarske učionice i jednak nivo prioriteta. One stoje na raspolaganju za rad sa korisnicima — učesnicima. Glavna radna stanica je namenjena nastavniku.

NAREDBE RADNE STANICE NASTAVNIKA

Postoje naredbe koje se mogu generisati samo sa radne stanice nastavnika. Postojanje ovih naredbi daje mogućnost da radna stanica nastavnika bude mali supervizor u sistemu. Prema tome iz kog režima se aktiviraju, naredbe ove grupe mogu se podeliti na:

A) Naredbe BASIC-a

— SPY n	— kontrola rada učenika (posmatranje ekrana radnih stanica)
— SEND	— slanje datoteke radnim stanicama (svima odjednom ili pojedinačno)
— MSG	— dozvola odnosno zabrana slanja poruka između radnih stanica

B) Naredbe EDITOR-a

— RUNLOCK	— dozvola čitanja svih datoteka
— RLOCK	— zabranjeno čitanje datoteka sa drugih kanala
— MSG	— dozvola slanja poruka (kao za MSG BASIC a)
— PUN LOCK	— dozvola korišćenja štampača
— PLOCK	— zabrana korišćenja štampača

NAREDBE KOJE SE AKTIVIRAJU SA BILO KOJE RADNE STANICE

A) Naredbe BASIC-a

— LOGIN	— uključenje u sistem RU
— LOGOUT	— isključenje iz sistema RU
— PGREAD	— čitanje programske datoteke sa diskete
— DREAD	— čitanje datoteke podataka sa diskete
— DWRITE	— upis programske datoteke na disketu
— MSG	— slanje poruka između radnih stanica
— LPRINT	— štampanje
— LLIST	— štampanje listinga programa

B) Naredbe EDITOR-a

— READ	— čitanje editovane datoteke sa diskete
— WRITE	— upis datoteke na disketu
— BWRITE	— upis bloka datoteke na disketu
— MSG	— slanje poruka između radnih stanica
— DIR	— prikazivanje imenika datoteke na ekranu
— REN	— promena naziva datoteke
— DEL	— brisanje datoteke
— COPY	— kopiranje datoteke
— PRINT	— štampanje

C) naredbe MONITOR-a

— T	— čitanje datoteke sa diskete
— Q	— upis datoteke na disketu

Dvokanalni UART	2×UART asinhroni serijski interfejs RS 232C
Flopi disk kontroler	za dvostruku flopi disk jedinicu 3 1/2 kapaciteta 2×1MB
Multiplex i jednokanalni UART	dodatni komunikacioni modul za spregu 16 radnih stanica sa centralnom jedinicom
Sistemski softver	DOS operativni sistem sa EDITOR-om, ASEMBLER-om i MONITOR-om
Programski jezici	BASIC (interpretator i kompilator), PLM 1800, μFORTH
2 Operaterski terminal	VIP 7251 RU ili emulator video terminala sa PECOM-om 64 i monitorom
3 Radne stanice	8 do 16 mikroračunara PECOM 64 sa monitorima. Dva režima rada: terminalski i autonomni
4 Serijski štampač	80—132 kolonski serijski sa interfejsom RS 232C



ØRIC NOVA 64

IDEALAN ŠKOLSKI RAČUNAR

RO NOVA
proizvodnja in servis p.o
Ljubljana, Titova 36
poštni predal 533/XI
telefon (061) 317-044
telex 31223 yu avtena

ORIC Nova 64 se svojim karakteristikama a posebno cenom izdvaja iz sivila konkuranata.

ORIC Nova je jednostavan po konstrukciji i u primeni a opet složen i pouzdan po onoma što pruža.

Kompjuter ORIC Nova 64 je na domaćim sajmovima i prezentacijama izazvao vanredno interesovanje, posteo popularan među mladim „kompjuterašima" i za kratko vreme se našao na listi najtraženijih u svojoj kategoriji.

Stručnjaci i test komisije koji su ga testirali dali su o njemu najlepše ocene i preporučili ga kao veoma pogodan računar za škole, pošto ispunjava sve uslove koje propisuje nastavni program.

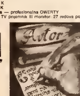
Lična karta ORIC Nova 64
CPU 6502A
ROM 16 K
RAM 64 K
Tastatura — profesionalna QWERTY
Slika — TV prijemnik ili monitor, 27 redova po 40/80 znakova, grafika 200×240 tačaka, 8 boja
Zvuk — tri odvojena kanala, raspon 7 oktava
Priključci — kasetofon, Centronika, ekspanzioni port za vezu sa diskaktnom jedinicom i drugim perifernim jedinicama
Softver — programi za obradu teksta, baze podataka, unakrsno izračunavanje, monitor, FORTH, školski softver

RO Nova za računar ORIC Nova 64 sprema više vrsta programa koje će vlasnici računara uskoro moći da nabave i to obrazovne programe, igre i klasične programe.

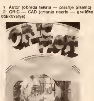
Preko Nove i njenih poslovnih partnera moći će da se kupe sledeći programi:

1 Autor (obrada teksta — pisanje pisama)
2 ORIC — CAD (crtanje nacrta — grafičko oblikovanje)
3 ORIC — MON (programiranje u mašinskom kodu)
4 ORIC — CALK (matematički program)
5 ORIC — FORTH (programski jezik)
6 Prošireni bejzik
7 Šah (logička igra)
8 Gorivo
9 Buboklop
10 Domine
11 Zid — dva različita programa
12 Froggy — dva različita programa
13 Snakes

Obrazovni programi
— Životinjski sistem
— Evolucija
— Periodni sistem
— Planete
— Sateliti
— Učenje Morzeovih znakova
— Hidroenergetski sistem SFRJ



NAPISATI PISMO?



... PROGRAMIRATI U MAŠINSKOM KODU?



... NACRTATI NACRTE?

5 ARTISTI NA TRAPEZU

Algoritmizacija zadataka
Milan Čabarkapa

Algoritmi

Čak i kada obavlja najtrivijalnija poslove, kao što su telefoniranje ili vožnja liftom, čovek se, navasno, koristi preciznim sistemom pravila koji se naziva algoritam. Pojam algoritma je poznat svakovima, ali poseban značaj dobija tek pojavom računara. Bez ovladavanja tehnikom građenja algoritama nema ni pravog programiranja. Sastavljanje algoritama je u velikoj meri stvaralački posao koji zahteva erudiciju i kreativnost, pa univerzalnog recepta nema, ali se strpljivim radom ipak može mnogo naučiti.

U nastavi matematike, fizike i gramatike učenici su već upoznali mnoštvo algoritama, mada ih možda nisu baš tako nazivali. I više od toga, zapazili su svakako da među mnogim pravilima s kojima se stalno susrećemo postoje ona koja propisuju niz dejstava kojima se postiže unapred zadati cilj.

Možemo kazati da svaki skup razumljivih i preciznih uputstava kako da se reši postavljeni zadatak iz bilo koje oblasti predstavlja algoritam, ali ako želimo da budemo do kraja korektni moramo dodati „u intuitivnom smislu".

Intuitivna definicija

Pod intuicijom podrazumevamo znanja stečena nekim dugotrajnim ogledom, ali još uvek nepodvrgnuta naučnoj analizi i stoga nedovoljno precizna i stroga. Kako ogled odmiče, tako se znanja obogaćuju, pa i naše intuitivne predstave mogu postepeno da se menjaju. Znanja u naučnoj formi ne smeju da budu tako promenljiva jer moraju da budu osnova daljih naučnih razmatranja. Međutim, onog trenutka kada formalizovana znanja prestanu da odgovaraju intuitivnim predstavama, te naučne formulacije moramo da zamenimo novim.

Na početku informatičkog obrazovanja obično je učenicima pojam algoritma intuitivno jasan, pa možemo posle niza primera da izložimo i formalnu definiciju ovog pojma. No, čini nam se da je korisnije i bolje ostaviti formalne definicije za kraj školovanja, a na početku se baviti praktičnim stranama problema algoritmizacije.

Izučavanje algoritama ima veliki metodski značaj. Učenik treba da otkrije važnost algoritama i njihov značaj za kreiranje računarskih programa. To što je za sve srednjoškolce postalo obavezno da steknu predstave o sastavljanju računarskih programa povlači da se kod njih formira određeni nivo algoritamske kulture. Doduše, pri definisanju ciljeva i zadataka predmeta Osnove tehnike i proizvodnje algoritamska kultura nije ni pomenuta, ali ako se uzme u obzir da su od 12 navedenih zadataka samo 2 u vezi sa informatikom i računarstvom na koje otpada 2/3 časova, vidi se da predlagačima ovog nastavnog programa i nije bilo sasvim jasno šta i zašto predlažu. Ta činjenica je još jedan prilog našoj tezi da bitan element opšte kulture, algoritamska kultura, skoro uopšte nije prisutna kod nas. Ova nevesela činjenica može se videti na svakom koraku — od planiranja rada, proizvodnje i investicija do planiranja za generaciju koja će početi da radi u 21. veku.

Pod algoritamskom kulturom podrazumevamo skup specifičnih predstava, umeća i navika potrebnih za ostvarivanje svrsishodne delatnosti i rešavanje složenih zadataka. Algoritamski pristup rešavanju zadataka i njihovo programiranje za izvršavanje na računaru zahteva od učenika preciznost i veću strogost rasuđivanja, što povlači povećanje naučnog nivoa rasuđivanja. Važno je napomenuti da pri praktičnom radu sa mikroračunarima ovi zahtevi ne dolaze spolja od nastavnika, već iz uzajamnog dejstva učenika i zadatka. Program koji sastavi učenik biva podvrgnut logici mašine koja prima i izvršava program upravo onako kako je napisan, a ne onako kako ga je zamislio njegov autor. Suprotstavljanje zamišljenog njegovoj realizaciji moćno je didaktičko sredstvo čije je obrazovne efekte teško i sagledati.

Uz to, algoritmizacija i navike programiranja aktiviraju razmišljanje i pogoduju razvoju kreativnosti. Zahvaljujući njima, kod učenika možemo postepeno i sistematično razviti takva važna opšteradna umeća kao što su umeće planiranje sopstvenog rada, racionalizacija rada i, što je možda najznačajnije, umeće sagledavanja efekata rada uz uzimanje u obzir realnih uslova njegove realizacije.

Pojam

Na prvi pogled bi se moglo pomisliti, kaže Knut na početku svoje trilogije „The art of computer programming", da je neko napisao „algoritam", nameravajući u stvari da kaže „logaritam", ali je permutovao prva četiri slova. Međutim, nije tako. Termin „algoritam" dobili smo kao latinski prevod imena uzbečkog matematičara i astronoma Al-Horezmije, Abu Ja'fer Mohammed Ibn Musa al-Khowarizmi, kako se u stvari zvao ovaj matematičar koji je živeo u IX veku, bavio se aritmetikom i algebrom i koristio je indijski sistem brojanja sa kojim su se preko prevoda njegovih dela u XII veku upoznali evropski matematičari. On je napisao znamenitu knjigu „Kitab al jabr w'al-muqabala", iz čijeg je naziva došlo i današnje „algebra", mada sama knjiga u stvari i nije bila sasvim algebarska.

U prvo vreme su se pod terminom algoritam podrazumevala pravila za vršenje četiri osnovne aritmetičke operacije: sabiranja, oduzimanja, množenja i deljenja. Kao što smo već rekli, danas ovom terminu odgovara daleko šire značenje. Pri tom treba imati u vidu da upotreba reči algoritam podrazumeva više od konačnog broja pravila kojima se zadaje niz operacija nužan za rešavanje klase problema, jer algoritmi imaju i neke karakteristične osobine.

i osobine algoritama

Šta mislite, da li pravilo „Kada izlaziš, ugasi svetlo" predstavlja algoritam? Da, mada vrlo primitivan. Ali pravilo „Za vreme kretanja trotoarom drži se leve strane" to nije, jer ima neprekidan karakter. Algoritmima možemo opisati jedino procese koji imaju diskretan karakter, koji se odvijaju kroz niz zasebnih dejstava. Ova dejstva opisuju se algoritamskim koracima koji se izvršavaju jedan za drugim. Svaki algoritamski korak sastoji se iz dva uputstva: šta treba uraditi i koji algoritamski korak treba izvršiti za njim. Dakle, algoritamski korak ne samo što precizira operaciju koju treba izvršiti, nego jednoznačno ukazuje i na sledeći algoritamski korak. Ovo svojstvo algoritama zove se **diskretnost**.

Algoritam treba da bude razumljiv za korisnika. Drugim rečima, svako uputstvo koje se zadaje algoritamskim koracima treba da bude takvo da ga izvršilac algoritma ume da uradi. Ako, recimo, tražite od učenika trećeg razreda osnovne škole da kvadrira neki broj, on vas verovatno neće razumeti, ali ako mu kažete da pomnoži broj sa samim sobom, moći će da reši zadatak. S druge strane, ako istu stvar tražite od srednjoškolca, nepotrebno je toliko razlagati problem.

Algoritam uvek mora da se završi posle konačnog broja koraka. Ovu osobinu zovemo **konačnost**. Pri tom, algoritam uvek mora da dovede do rešenja zadatka. Primetimo da i utvrđivanje činjenice da zadatak nema rešenje predstavlja u stvari rešenje problema. Ovo svojstvo algoritama zove se **rezultativnost**.

Algoritmi imaju još jednu važnu osobinu — **masovnost**. To svojstvo podrazumeva da se pomoću jednog istog algoritma može rešavati čitava klasa zadataka sa raznim ulaznim veličinama. Svojstvo masovnosti značajno uvećava praktičnu vrednost algoritama. Uz diskretnost, rezultativnost, efektivnost i masovnost, ono što suštinski određuje algoritme je svojstvo **determinisanosti**. Naime, za iste ulazne veličine algoritam uvek mora dati iste rezultate.

Kreiranje algoritama

Kreiranju algoritama posvećujemo metodičku zbirku od pedeset izabranih zadataka. Nadamo se da zbirka zadataka iz algoritmizacije iskusnog pedagoga Milana Čabarkape daje solidnu osnovu za sticanje algoritamske kulture. U njoj su kao primeri rešeni zadaci značajni za dalji rad u oblasti programiranja, a za karakteristične probleme prodiskutovano je više rešenja. Rešenja su data tako da se mogu programirati u bilo kom višem programskom jeziku. Zahvaljujući tome, tekst koji sledi biće korisan kako onima koji računarsku pismenost uče na bejziku, tako i učenicima koji počinju od paskala.

Sastavljanje algoritma

Građenje algoritma je u velikoj meri stvaralački proces koji često zahteva veliku erudiciju i kreativnost i stoga je nemoguće dati neke opšte recepte. Za početak uzmimo jednostavan zadatak čija ćemo rešenja izraziti algoritamski.

PRIMER 1. Za učenike koji su zaposleni preko omladinske zadruge izračunati bruto (u oznaci B) i neto (N) dohodak ako je poznat broj radnih sati (S) cena po satu (C) i procenat odbijanja (P) na osnovu određenih doprinosa.

Da bi se rešio ovaj zadatak treba odgovoriti na sledeća pitanja

1 Koja su veličina poznate a koje nisu?

2 Koje su moguće veze između poznatih i nepoznatih veličine? Drugim rečima treba odabrati najpogodniji niz operacija nad poznatim veličinama da bi se došlo do željenih rezultata.

3 Šta se izdaje kao rezultat?

Odgovori

1 Poznate veličina su S, C i P, a nepoznate B i N

2 Između poznatih i nepoznatih veličina postoje sledeće veze

B=S•C

N=B−B•P/100

3 Rezultat su vrednosti dodeljene B i N

Seda se algoritam može izraziti na sledeći način.

Korak 1 Definisati vrednosti za ulazne veličine S C P

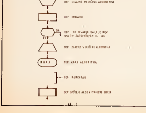
Korak 2 Izračunati bruto dohodak B=S•C

Korak 3 Izračunati neto dohodak N=B−B•P/100

Korak 4 Definisati izlazne veličine B i N

Algoritamske sheme

Pošto je tekstualni opis algoritma u složenijim zadacima teško čitljiv, a samim tim mukotrpno je i traganje za greškama programeri se služe znatno preglednijim grafičkim prikazom algoritma — algoritamskom shemom. Na slici 1 prikazani su grafički simboli koji se po konvenciji koriste za pojedine algoritamske korake



sl. 1

Sada nije teško sastaviti algoritamsku shemu (slika 2) za prethodni primer obračuna bruto i neto dohotka



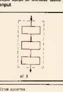
sl 2

Veličinu kojoj se vrši dodela treba pisati na levoj strani znaka jednakosti u protivnom upotrebiti znak =>

Osnovne strukture od kojih se sastoji ma koje algoritamska shema su linijska, razgranata cikilčka

Linijske strukture

Kod LINIJSKE ALGORITAMSKE STRUKTURE (slika 3) svaki algoritamski korak se izvršava tačno jedanput.



sl 3

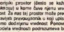
Prema tome algoritam primera 1 ima linijsku strukturu

> ### Mućni malo glavom
>
> 1 Sastaviti algoritamsku shemu kojom se za zadatu vrednost temperature u Farenhajtovim stepenima preračunava u Celzijusova
>
> Uputstvo C=(F−32)•5/9
>
> 2 Sa punim automobilskim rezervoarom kapaciteta V litara pređen je put od S kilometara. Sastaviti algoritamsku shemu kojom se računa potrošnja u litrima na 100 kilometara
>
> 3 Sastaviti algoritamsku shemu po kojoj se D američkih dolara preračunava u M nemačkih maraka ako su poznati kurs dolara X i marke Y

Vrlo je važno pri upoznavanju sa linijskim shemama usvojiti algoritam za zamenu vrednosti promenljivih da se na tome ne bismo kasnije spoticali pri izradi složenijih algoritama (na primer sortiranja)

PRIMER 2 Sastaviti algoritamsku shemu kojom se po dodeli ulaznih vrednosti promenljivim A i B razmenjuju dodeljena vrednosti, a zatim izdaju nove vrednosti za obe promenljive

Ovde promenljiva predstavlja simboličku oznaku kojoj se pridružuje određena vrednost. Za svaku promenljivu računar rezerviše memorijski prostor (često se kaže i lokaciju) u kome čuva njenu vrednost. Za nas taj prostor može predstavljati pravougaonik u koji upisujemo dodeljenu vrednost. Ponovna dodela vrednosti podrazumeva brisanje zatečene



sl 4

Zadatak ćemo rešiti uvođenjem pomoćne promenljive P kojoj ćemo najpre dodeliti vrednost promenljive A (korak 1 slika 4 i slika 5) Dodela podrazumeva kopiranje vrednosti od A, čime se ne gubi vrednost ove promenljive. Zatim možemo slobodno vrednost promenljive B dodeliti promenljivoj A (korak 2) Vrednost P (sačuveno A) dodeliti promenljivoj B (korak 3)

Kome je ovo lako neka se pozabavi malom mozgalicom *kako razmeniti vrednosti promenljivih A i B bez pomoćne promenljive*



sl 5

Razgranate strukture

RAZGRANATE ALGORITAMSKE STRUKTURE (slika 6) sadrže algoritamski korak u kome se ispituje da li je neki uslov zadovoljen ili ne i na osnovu toga se nastavlja jednom od dva moguće grane. U ovakvoj algoritamskoj strukturi svaki algoritamski korak se izvršava najviše jedanput



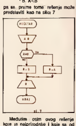
sl 6

PRIMER 3 Sastaviti algoritamsku shemu kojom se izračunava maksimalna vrednost M od dve zadate brojne vrednosti A i B, to jest M=max (A,B)

Ovo se može formulisati kao

$$M=\begin{cases} A & A>=B \\ B & A<B \end{cases}$$

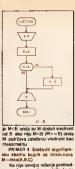
pa se prema tome rešenje može predstaviti kao na slici 7



sl. 7

Međutim osim ovog rešenja koje je najprirodnije i koje se od početnika može očekivati korisno je da se usvoji rešenje sa slike 8 koje predstavlja blagi uvod u algoritam za određivanje maksimalne vrednosti između 3, 4 ili proizvoljno mnogo zadatih vrednosti

Dakle, pretpostavili smo da je A maksimalna vrednost i prema tome M dodeljujemo vrednost od A. Sledećim korakom se proverava da li je pretpostavka održiva to jest ako

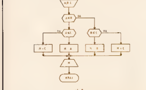


sl 8

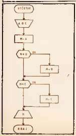
je M<B onda se M dodeli vrednost od B ako nije M<B (M>=B) onda M zadržava zatečenu vrednost kao maksimalnu

PRIMER 4 Sastaviti algoritamsku shemu kojom se izračunava M=max(A,B,C)

Ko nija usvojio rešenje prethodnog primera najverovatnije će sedetak rešiti kao na slici 9



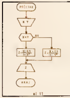
sl 9



sl 10

Najpri rodniji način razmišljanja međutim ne mora uvek da bude najpregledniji najelegantniji i najpodesniji za izradu programa i izvršavanje na računaru U to nas uverava rešenje prikazano na slici 10, koje se zasniva na drugoj ideji iz prethodnog zadatka.

Za određivanja minimalna vrednosti dva ili tri elementa potrebno je samo u uslovnim algoritamskim koracima prethodnih rešenja promeniti znak.



sl 11

PRIMER 5 Sastaviti algoritamsku shemu kojom se sa zadata X i Y izračunava Z po sledećoj formuli

$$Z=\frac{\min(X,Y)+9.5}{1+\max^2(X,Y)}$$

U ovom zadatku nema potrebe da se posebno određuju min i max, pa da se onda stave u formulu Dovoljno je da se jednim uslovom ustanovi da li je veće X ili Y (slika 11)

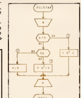
PRIMER 6 Sastaviti algoritamsku shemu kojom se sa zadata X izračunava Y po formuli

$$Y=\begin{cases} X^2-3, & X<0 \\ 0, & 0\leq X\leq 1 \\ X^2+X, & X>1 \end{cases}$$



sl 12

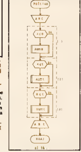
Prema osnovnoj formuli, realna prava se razbija na tri intervala (slika 12) i u svakom intervalu se računa po naznačenoj podformuli Idući od levog intervala dolazimo do rešenja na slici 13



sl. 13

PRIMER 7 Sastaviti algoritamsku shemu kojom se zadaju vrednosti promenljivih A B i C, a zatim sa premeštanjem dodeljenih vrednosti dovode u monotono neopadajući poredak (Y≤B≤C)

Osnovna ideja se sastoji u toma da se najpre porede A i B zatim A i C i na kraju B i C Ako su dva broja uređena na željeni način ide se na sledeće poređenja ako nisu razmenjuju vrednosti. Razmena se realizuje algoritmom iz primera 2 Dakle iz prva dva poređenja u A se smesti najmanja vrednost između A B i C, trećim poređenjem se obezbeđuje da sledeća vrednost po veličini bude u B, a najveća u C (slika 14)



sl 14

Da ne bismo dobili preglomaznu algoritamsku shemu logičke celine kojima se realizuje razmena vrednosti promenljivih A i B A i C i B i C možemo predstaviti opštijim korakom Na primer razmena promenljivih A i B sa slike 15 se



sl 15

realizuje prema primeru 2 kao na slici 16



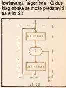
sl 16

Korektnost algoritma možemo testirati primerom

A	B	C	
3	2	1	Ulazna vrednosti
2	3	1	Rezultat posle bloka (1)
1	3	2	Rezultat posle bloka (2)
1	2	3	Rezultat posle bloka (3)

Ciklične algoritamske strukture

Cikličke algoritamske strukture sadrže niz algoritamskih koraka koji se više puta ponavlja u toku izvršavanja algoritma Ciklus opšteg oblika se može predstaviti kao na slici 20



sl 20

Vrlo često algoritamske shema sadrže samo jedan od dva niza algoritamskih koraka (ili gornji ili donji) Ciklus koji ne sadrži donji niz koraka (uslov na kraju ciklusa) naziva se **iterativni.** Njegov niz naredbi se izvršava bar jednom Ako ciklus ne sadrži gornji niz koraka naziva se **ciklus po uslovu.** Ima još jedan tip ciklusa za koji u programskim jezicima postoja odgovarajuće naredbe (u bejziku FOR u fortranu DO) To je **prebrojivi** ciklus koma je broj ponavljanja poznat unapred, pre prvog prolaska kroz ciklus

Prebrojivi ciklus sadrži **ciklusnu** (ili kontrolnu) **promenljivu, početnu vrednost** ciklusne promenljive **priraštaj** za koji se uvećava i **krajnju vrednost** od koje ciklusna promenljiva do izlaska iz ciklusa, ne sme biti veća (u slučaju pozitivnog priraštaja) odnosno manja (u slučaju negativnog priraštaja). Počet-

Mućni malo glavom

1 Sastaviti algoritamsku shemu kojom se izračunava Z na osnovu formula

a)

$$Z=\text{abs}(x)=\begin{cases} x & x>0 \\ -x & x<0 \end{cases}$$

b)

$$Z=\text{sgn}(X)=\begin{cases} -1 & X<0 \\ 0 & X=0 \\ +1 & X>0 \end{cases}$$

c)

$$Z=\begin{cases} \min(X,Y) & Y\geq 0 \\ \max(X^2,Y^2) & Y<0 \end{cases}$$

2 Sastaviti algoritamsku shemu kojom se izračunava Y po formuli

$$Y=\begin{cases} X^2+1 & -1<X<4 \\ X^3-X & 5<X\leq 6 \\ 1/X & \text{u ostalim slučajevima} \end{cases}$$

3 Periodična funkcija perioda 1 definisana je za realne brojeve X. Grafik na [0,1] prikazan je slici 17

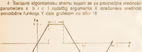


sl 17

Sastaviti algoritamsku shemu kojom se izračunava vrednost funkcije Y za zadatu vrednost argumenta X

UPUTSTVO Transformacijom X−[X] (gde [X] znači ceo deo od X) računanje se svodi na interval [0,1]. Uzeti da je dopuštena upotreba funkcija ceo deo

4 Sastaviti algoritamsku shemu kojom se za proizvoljne vrednosti parametara a, b i c i zadatog argumenta X izračunava vrednost periodične funkcije Y date grafikom na slici 18

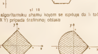


sl 18

5 Sastaviti algoritamsku shemu kojom se ispituje da li tačka sa koordinatama (X,Y) pripada šrafiranoj oblasti

a) Na slici 19a

b) Na slici 19b



sl 19

UPUTSTVO Pokazati da tačka pripada šrafiranoj oblasti ako koordinate zadovoljavaju uslov

a) $|X|+|Y|<a$ b) $X^2+Y^2<a^2$, $X+|Y|<a$

na i krajnja vrednost kao i priraštaj moraju biti poznati pre prvog prolaska kroz ciklus. Neka je

I — ciklusna promenljiva,

P — početna vrednost ciklusne promenljive

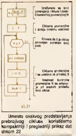
K— krajnja vrednost ciklusne promenljive,

S — priraštaj ciklusne promenljive

Tada, ako je K>P, priraštaj S mora biti pozitivan, ako je K<P priraštaj mora biti negativan. Broj ponavljanja ciklusa daje sledeća formula

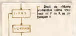
$$N=\left[\frac{K+S-P}{S}\right]$$

Princip rada prebrojivog ciklusa daje algoritamska shema na slici 21



sl. 21

Umesto ovakvog predstavljanja prebrojivog ciklusa koristićemo kompaktniji i pregledniji prikaz dat slikom 22

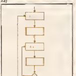


sl 22

Ako je priraštaj 1 može se izostaviti. Primer: ako u shemi imamo da je I=2, 9, 2 ciklus se izvršava uz vrednosti ciklusne promenljive I=2, 4, 6, 8. Ako je zapisano I=3.5, 2.0, −0.5, ciklus se izvršava uz vrednosti ciklusne promenljive I=3.5, 3.0, 2.5, 2.0

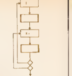
Pri upotrebi prebrojivih ciklusa treba imati na umu sledeće:

1 Dozvoljena je koncentrična kompozicija dva i više ciklusa (slika 23)



sl 23

2 Nije dozvoljeno presecanje ciklusa, iako će neki interpretatori prihvatiti tako napisan program (slika 24)



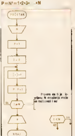
sl 24

3 Dozvoljen je izlaz iz ciklusa ukoliko je neki uslov zadovoljen. Posle izlaza ciklusna promenljiva ima vrednost za priraštaj veću od one koju je imala u poslednjem izvršavanju naredbi tela ciklusa

4 Skok u ciklus nije dozvoljen, mada će odgovarajući neregularan program raditi

Primer 8. Sastaviti algoritamsku shemu kojom se za zadato N(≥1) izračunava

$P=N!=1\cdot2\cdot3\cdot\ldots\cdot N$



Računanje se realizuje iz N koraka. Promenljiva P redom uzima vrednosti:

1

1·2

1·2·3

. . . . .

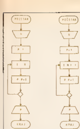
1·2·3· … ·N

Poslednja izračunata vrednost je izlazna (slika 25)

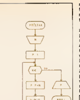
Pošto je ovde unapred poznat broj prolaza kroz ciklus ekvivalentni algoritmi se mogu dobiti upotrebom prebrojivog ciklusa sa brojanjem Unapred (slika 26) ili unazad (slika 27) ili unazad računajući i 0! (slika 28)

Primer 9 Sastaviti algoritamsku shemu kojom se za zadati prirodan broj N izračunava

$$S=1+\frac{1}{2}+\ldots+\frac{1}{N}$$



sl 26 sl 27



sl 28

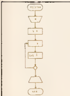
Suma N sabiraka se obično računa u N koraka kao rezultat izvršavanja I-tog koraka dobija se suma prvih I sabiraka Primenom ovog metoda (slika 29) promenljiva S uzima vrednosti

$$1$$

$$1+\frac{1}{2}$$

$$1+\frac{1}{2}+\frac{1}{3}$$

$$1+\frac{1}{2}+\frac{1}{3}+ \quad +\frac{1}{N}$$

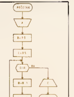

sl 29

PRIMER 10 Sastaviti algoritamsku shemu kojom se među brojevima

0 5, 0 5+1,5, 0 5+1 5+2 5,

pronalazi prvi veći od datog broja A

Izvršavanjem algoritma (slika 30) promenljiva B uzima vrednosti 0 5 1 5 2 5 , promenljiva C uzima vrednosti 0 5 0 5+1 5 0 5+1 5+2 5 Ove vrednosti se formiraju sve dok se ne dobije vrednost promenljive C koja je veća od vrednosti promenljive A Pred ulazom u ciklus promenljive B i C dobijaju početnu vrednost 0 5

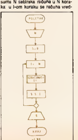


sl 30

PRIMER 11 Sastaviti algoritamsku shemu kojom se izračunava suma

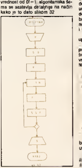
$S=1!+2!+3!+ \dots +N!$

U primeru 9 smo videli da se suma N sabiraka računa u N koraka u I-om koraku se računa vrednost I-og sabirka i dodaje sumi već formiranih I-1 sabiraka. Prema tome, grubu shemu možemo postaviti kao na slici 31



sl 31

Zbog složenosti algoritma i brzina izvršavanja nije preporučljivo pri svakom prolazu kroz ciklus primenjivati algoritam za računanje I!. Ako važi da je $I!=(I-1)!\cdot I$, $0!=1$ tada u I-tom prolazu kroz ciklus vrednost I! se računa množenjem izračunatog faktorijela u prethodnom prolazu kroz ciklus sa I Ako za računanje faktorijela uvedemo promenljivu F koja ima početnu vrednost od $0!=1$ algoritamska šema se sastavlja detaljnije na način kako je to dato slikom 32



sl 32

Zadaci sa nizovima

Do sada smo zadatke rešavali isključivo upotrebom prostih promenljivih za čuvanje određenih vrednosti Međutim ogroman je broj zadataka koji bi se upotrebom prostih promenljivih vrlo teško rešio, ili se ne bi mogao uopšte rešiti Rešavanje takvih zadataka olakšava upotreba nizova

Niz je skup podataka sa zajedničkim imenom Element niza se naziva indeksna promenljiva Indeksna promenljiva se piše navođenjem imena niza i para zagrada unutar kojih se pišu celobrojni indeksi razdvojeni zarezom Prema broju indeksa nizovi mogu biti jednodimenzionalni dvodimenzionalni i višedimenzionalni

Uvek treba imati na umu da upotreba niza na primer A(1) A(2) ... A(50) podrazumeva rezervisanje prostora (u računaru potrebnog broja registara) za čuvanje tekućih vrednosti indeksnih promenljivih Na primer

Z1

A1

A

Ako se u zadatku koristi 100 brojnih vrednosti onda se može uvesti indeksiranje

B(1), B(2) ,... B(100),
ili B(0) B(1) ... B(99),
ili B(−3) B(−2) ... B(96),

Mućni malo glavom

1 Sastaviti algoritamsku shemu kojom se među brojevima

$$1 \qquad 1+\frac{1}{2} \qquad 1+\frac{1}{2}+\frac{1}{3}$$

pronalazi prvi veći od zadatog broja

2 Sastaviti algoritamsku shemu kojom se ... brojeve M i N izračunava

$$S=N\,(N+M)\,(N+2M) \quad (N+M\cdot)$$

3 Sastaviti algoritamsku shemu za ... e

$$S=\frac{1}{3^2}+\frac{1}{5^2}+ \quad + \quad \frac{1}{(2N+1)^2}$$

ako je zadat prirodan broj N

4 Sastaviti algoritamsku šemu kojom se ... izračunava

$$S=\frac{1!}{\frac{1}{2}}+\frac{2!}{\frac{1}{2}+\frac{1}{3}}+ \quad + \quad \frac{}{\frac{1}{2}}$$

5 Sastaviti algoritamsku shemu kojom se ... zadat ... izračunava

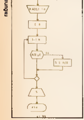
a) $S=3!-6!+9!- \quad +(-1)^N \; (3N)$

b) $S=1\cdot2+2\cdot3\cdot4+ \quad +N\cdot(N \quad 1) \quad (2$

Ili bilo koji siričan način indeksiranje teoretski može ići od bilo kog celog broja pa u odgovarajućem programskom jeziku traba proveriti ovu mogućnost. Indeks se uvećava za jedan pri prelazu na sledeći element niza. Ako se u shemi koristi niz čije su indeksna promenljive B(0), B(1), ..., B(99), a prosta promenljiva I uzima vrednosti od 0 do 99, to se može slobodno pisati B(I). Promenom vrednosti proste promenljive I „poziva" se odgovarajuća indeksna promenljive B(I).

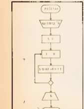
PRIMER 12. **Sastaviti algoritamsku shemu kojom se za zadato N, izračunava suma pozitivnih elemenata niza A(1), A(2), ..., A(N).**

Suma se računa iz N koraka. Rezultat izvršenja I-tog koraka predstavlja rešenje zadatka za deo niza koji sadrži elemente A(1), A(2), ..., A(I). Ciklusna promenljiva se koristi za indeksiranje niza (slika 33).



sl 33

PRIMER 13. **Sastaviti algoritamsku shemu kojom se za zadate nizove A(1), A(2), ..., A(N) i B(1), B(2), ..., B(N) pri datom prirodnom broju N izračunava**



sl 34

$$S = A(1)*B(N) + A(2)*B(N-1) + \dots + A(N)*B(1)$$

Iz sheme na slici 34 se vidi da je ciklusna promenljiva iskorišćena tako da obezbeđuje formiranje članova sume

A(1)*B(N), A(2)*B(N−1) ... A(N)*B(1) kada se I kreće od 1 do N.

PRIMER 14. **Sastaviti algoritamsku shemu kojom se određuje maksimalna vrednost koju imaju elementi niza A(1), A(2), ..., A(N).**

Maksimalna vrednost se određuje iz N koraka: po izvršenju I-tog koraka vrednost promenljive M jednaka je vrednosti najvećeg elementa između prvih I elemenata niza.

Prvim korakom se promenljivoj M dodeljuje vrednost prvog elementa, s pretpostavkom da je to najveći element niza. Svaki sledeći korak je provera da li vrednost promenljive maksimuma M ostaje ili se uvećava ako se naišlo na element niza veći od nje. Videti sliku 35.



sl 35



sl 36

PRIMER 15. **Sastaviti algoritamsku shemu za formiranje niza A(1), A(2), ..., A(100), čije su vrednosti B(1), C(1), B(2), C(2), ..., B(50), C(50), ako su dati nizovi B(1), B(2), ..., B(50) i C(1), C(2), ..., C(50).**

Formiranje niza se realizuje u 100 koraka. U I-tom koraku se dodeljuje odgovarajuće vrednost promenljivoj A(I). Dodela se vrši zavisno od parnosti I: ako je I=2*M, tada je A(I)=C(M), ako je I=2*M−1, tada je A(I)=B(M). Videti sliku 36.

PRIMER 16. **Sastaviti algoritamsku shemu kojom se realizuje cikličko premeštanje vrednosti elemenata niza A(1), A(2), ..., A(N) za jedno mesto ulevo.**

Traženo premeštanje (slika 37) se realizuje tako što vrednost A(1) sklonimo u pomoćnu promenljivu P, a zatim u N−1 koraka vrednosti premeštamo ulevo kroz dodele

A(1)=A(2)   A(2)=A(3)
A(N−1)=A(N)

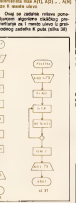
Dodelom A(N)=P sačuvanu vrednost od A(1) šaljemo u A(N)

2   3   N

A(1) A(2) A(3) ... A(N−1), A(N)

P

1   N+1



sl 37

PRIMER 17. **Sastaviti algoritamsku shemu kojom se realizuje cikličko premeštanje vrednosti elemenata niza A(1), A(2), ..., A(N) za K mesta ulevo.**

Ovaj se zadatak rešava ponavljanjem algoritma cikličkog premeštanja za 1 mesto ulevo iz prethodnog zadatka K puta (slika 38).

PRIMER 18. **Sastaviti algoritamsku shemu kojom se za zadato X računa beskonačna suma**

$$S = 1 - \frac{X}{1!} + \frac{X}{2!} - \dots + (-1)^N \frac{X^N}{N!} + \dots$$

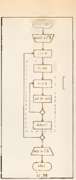
Sumirati do člana koji je po apsolutnoj vrednosti manji od zadatog E.

Ako članove sume označimo sa
A(0)=1

$$A(1) = -\frac{x}{1!} \quad A(1) = -\frac{x}{1!}$$

$$A(2) = \frac{x^2}{2!},$$

$$A(N) = (-1)^N \frac{x}{N!},$$



sl 38

to se može uzeti količnik dva susedna člana niza

$$A(I+1)/A(I) = \frac{(-1)^{I+1}\frac{x^{I+1}}{(I+1)!}}{(-1)^I \frac{x^I}{I!}} = \frac{-x}{I}$$

koji omogućuje da se članovi niza računaju na osnovu prethodno izračunatih vrednosti. Dakle

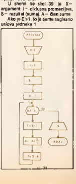
$$A(0)=1 \quad A(I+1) = \frac{-A(I)*x}{I}$$

I=0,1,2,

je veza kojom se izbegava ponavljanje računanja.

U shemi na slici 39 je X — argument, I — ciklusna promenljiva, S — rezultat (suma), A — član sume.

Ako je E>1, to je suma saglasno uslovu jednaka 1.



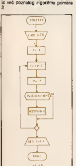
sl 39

PRIMER 19 Sastaviti algoritamsku shemu kojom se niz međusobno različitih elemenata A(1), A(2) ..., A(N) uređuje testerasto A(1) A(2)>A(3) A(4)>

Željeno uređenje (slika 40) postiže se iz N−1 koraka naizmeničnim postavljanjem susednih elemenata u poredak

A(1)<A(2) A(2)>A(3) A(3)<A(4)

Promenljiva P uzimajući vrednosti 1 ili −1 reguliše odgovarajući uslov u svakom prolazu kroz ciklus Opštij: korak A(I)⇄A(I+1) se izvodi iz već poznatog algoritma primera 2



sl 40

PRIMER 20 Sastaviti algoritamsku shemu kojom se brojni niz A(1), A(2), A(N) uređuje u monotono neopadajući poredak, to jest

A(1)<A(2)<A(3)<...<A(N)

Zadatak ćemo rešiti na način (slika 41) koji je verovatno očigledniji u odnosu na druge ali nije i najoptimalniji

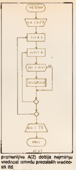
Ideja algoritma je slična ideji iz primera 7 za uređenje samo tri broja Ideja za uređenje se može ilustrovati na sledeći način

prvi prolaz kroz niz

A(1) A(2) A(3) A(4) A(N)

drugi prolaz kroz niz

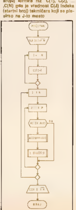
Pri prvom prolazu kroz ciklus po I(=1) porede se vrednosti indeksne promenljive A(1) sa A(2), A(3) A(N) i zamenjuju se vrednosti tamo gde A(1) nije manje ili jednako od navedenih vrednosti Dakle posle prvog prolaza kroz ciklus po I u A(1) će se postaviti najmanja vrednost iz niza, a vrednost od A(1) ukoliko nije najmanja biće premeštena u neku indeksnu promenljivu A(2) A(3) A(N) U drugom prolazu kroz ciklus po I(=2) istim postupkom indeksna



sl 41

promenljiva A(2) dobija najmanju vrednost između preostalih vrednosti itd

PRIMER 21 Sastaviti algoritamsku shemu kojom se na osnovu N različitih rezultata trkača na 100 m datih nizom A(1), A(2), ,A(N) (indeks odgovara startnom broju) formira niz C(1), C(2), ,C(N) gde je vrednost C(J) indeks (startni broj) takmičara koji se plasirao na J-to mesto

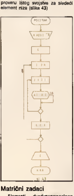


sl 42

Ideja za rešavanje ovog zadatka sastoji se u tome da se uvede niz C(1) C(2), C(N) čije su vrednosti redom 1,2 3, N to jest startni brojevi takmičara sa rezultatima A(1) A(2) A(N) Rezultati A(1) A(2) ,A(N) uređuju se u monotono neopadajući niz pri zameni vrednosti elemenata ovog niza ostvaruje se i zamena vrednosti elemenata niza C(1) C(2) C(N) koji imaju iste indekse Dakle startni broj prati premeštanje svog rezultata Na taj način indeksne promenljive C(1) C(2) C(N) dobijaju vrednosti startnih brojeva u poretku od najboljeg do najlošijeg rezultata koji je dat uređenjem niza A(1) A(2) ,A(N) Videti sliku 42

PRIMER 22 Sastaviti algoritamsku shemu kojom se određuje broj različitih elemenata niza A(1), A(2), ,A(N)

Za svaki element od 2-og do N-tog (i=2 N) se proverava da li među elementima ispred njega (J=1 2, I−1) ima njemu jednakih Ako takvih elemenata nema broja C se uvećava za 1 Ako je jednak bar jednom od elemenata koji mu prethode prelazi se na proveru istog svojstva za sledeći element niza (slika 43)



sl 43

Matrični zadaci

Elementi dvodimenzionalnog niza, npr A(I,J) gde je I=1,2 N ,a J=1,2, ,M najčešće se radi preglednosti predstavljaju matrično (tablično)

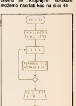
A(1,1) A(1,2) A(1 M)
A(2,1) A(2,2) A(2 M)
...
A(N,1) A(N,2) A(N M)

PRIMER 23 Sastaviti algoritamsku shemu kojom se od zadatih nizova A(1), A(2) ,A(M) i B(1), B(2), ,B(N) formira matrice Anxm (n − broj vrsta, m broj kolona) čiji su elementi

$$C(I,J)=\frac{A(J)}{1+B(I)} \quad I=1,2 \quad ,N, \quad J=1\,2 \quad ,M$$

Algoritam ćemo organizovati tako da najpre formiramo elemente prve vrste matrice C zatim druge itd

Prema tomme, algoritamsku shemu sa krupnijim korakom možemo nacrtati kao na slici 44

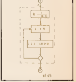


sl 44

Zadatak smo sveli na podzadatak koji se sastoji u tome da za zadatu vrednost od I (1⩽I⩽N) treba promenljivima C(I 1) C(I 2) C(I M) redom dodeliti vrednosti

$$\frac{A(1)}{1+B(I)} \quad \frac{A(2)}{1+B(I)} \quad \frac{A(M)}{1+B(I)}$$

To se može realizovati kao na slici 45



sl 45

Dodatna promenljiva Q je upotrebljena da bi se izbegla ponavljanja već izvršenih izračunavanja Sada se može sastaviti krajnja shema (slika 46)

Mucni malo glavom

1 Sastaviti algoritamsku shem ... A(200) formiraju nizovi B(...) ... C(100) čiji su elementi redom jednak ... A(1) ... A(200)

2 Sastaviti algoritamsku shem ... cna znaka u nizu A(1) A(2) ... A(N) ele ...

3 Sastaviti algoritamsku shemu ... A(2) ... A(N) premeštaju i inverzan poreč ...

4 Sastaviti algoritamsku shemu ko ... itivni element u nizu A(1) A(2) ... A(N)

5 Sastaviti algoritamsku shemu k ... B(M) izbacivanjem iz zadatog niz ... elementa i svih njemu jednakih

6 Sastaviti algoritamsku shemu k ... A(N) zdvaja n z međusobno različitih ele ...

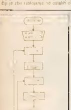
Sastaviti algoritamsku shem ...

$$1 + ax + \frac{a(a-1)}{2} x^2 + \ldots$$

Računat do člana sume koj je ... apsolutnoj ... od zadatog E la je realan broj

8 Nizom (X(I),Y(I)) I 1 ... N date su loka ... u nogosagrađenom naselju Svak objekt n a prizemi ... edvi dene za neki od pratećih objekata (sa uslug ... bu tik ... Sastaviti algoritamsku shemu kojom se ... objekt (njegov indeks) koj je najpovoljniji za otvaranje ... jest čiji je zbir rastojanja od ostalih objekata naj ...



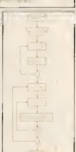


sl 46

PRIMER 24 Data je matrica A_{nxn}. Sastaviti algoritamsku shemu za formiranje niza D(1) D(2) ... D(N), gde je D(1) suma elemenata na glavnoj dijagonali, a D(J) (J=2, ..., N) suma elemenata J-te dijagonaline paralele u donjem trouglu matrice

Elementima niza D(1) ... D(N) se dodeli početna vrednost 0. Prolazom kroz donji trougao matrica (uključujući i dijagonalu) element A(I,J) se dodeljuje sumi koja se formira posredstvom indeksne promenljive D(I−J+1). Videti sliku 47

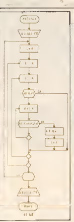
Primer 25 Data je matrica A_{nxn} čiji su elementi



1 ako postoji direktan jednosmeran put od grada I do
A(I,J)= grada J
0 u suprotnom

Sastaviti algoritamsku shemu kojom se matrica A transformiše u matricu iz koje se vidi koji su gradovi u vezi bez obzira na broj presedanja

Prolaz kroz matricu organizovan ciklusima po I i J tamo gde nije ustanovljena veza između gradova I i J, tj. A(I,J)=0, proverava da li se može ostvariti veza preko nekog od gradova K (K=1, ... N). Ako može nova veza se evidentira dodelom A(I,J)=1. Indikator L treba da pokaže da li je prolazom kroz matricu ustanovljena neka nova veza između gradova. Ako jeste (L=1) treba nastaviti traganje za mogućom vezom, ako nije (L=0) treba prekinuti, jer nije došlo do promena u matrici, odnosno ne mogu se ustanoviti nove veze (slika 48)



sl 48

Mučni malo glavom

1 Sastaviti algoritamsku shemu kojom se za matricu Anxn izračunava

a) suma svih elemenata

b) sume dijagonalnih elemenata, to jeste elemenata sa jednakim indeksima

c) vrednost najvećeg i najmanjeg elementa na sporednoj dijagonali,

d) sume elemenata ispod glavne dijagonale

e) sume pozitivnih i sume negativnih elemenata

2 Sastaviti algoritamsku shemu kojom se na osnovu zadate matrice Anxn formira niz B(1) B(2) ... B(N) čiji su elementi redom jednaki

a) sumi elemenata po vrstama matrice A

b) maksimalnoj vrednosti po kolonama matrice A

c) 1 ako su elementi vrste uređeni u monotono neopadajućem poretku 0 u suprotnom

3 Sastaviti algoritamsku shemu kojom se zamenjuju vrednosti ... zadate matrice Anxn

* 4 Sastaviti algoritamsku shemu kojom se kolone matrice Anxn uređuju u monotono nerastući poredak

MALI RAČUNARI, VELIKO ZADOVOLJSTVO, **ATARI** VRHUNSKA TEHNOLOGIJA, PRISTUPAČNE CENE

ATARi 520 stm+FLOPPY SF 314+MIŠ=1.009.- DM

—Lični računar, kojeg možete priključiti ne TV prijemnik!



— mikroprocesor 16/32 bit/Motorola 68000/8 MHz, 512 KB RAM, 192 KB ROM
— IBM kompatibilan!!!

Sva cena su informativne!
TELEFONIRAJTE NAM NA TELEFDNSKI BRDJ
061/327-641, 327-643

mladinska knjiga

KOPRODUKCIJA 61000 LJUBLJANA
Prešernova 5 ZASTUPSTVO ATARI
Cigaletova 6
OOBIĆETE DOPUNSKE INFORMACIJE ILI PREDRAČUN PO VAŠOJ ŽELJI!

U našem programu imamo i „8-bitni računar " ATARI 130 XE



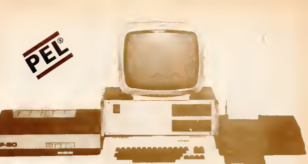
— ATARI 130 XE	=285 DM
— kaeetofon	=90 DM
— palice za igrenje	=16 OM

Oeviznim cenema morate dodati i oko 70% dinarskih dažbina!

OMOGUĆITE SVOJOJ DECI, DA POZNAVANJEM RADA SA RAĆUNAROM OSIGURAJU MESTD U VREMENU KOJE OOLAZI.

PDSEBNA POGDDNDST ZA ŠKOLE!
Prodaja ATARI računara 520 STM i za dinare. Ze povoljnu cenu đobićete snažan računar, za kojeg vam nudimo besplatan software paket (IBM kompatibilnost, grafički progra mi, programi za uređivanje teksta i dr.)

POSEBNA POGODNOSTI
NA ZALIHI IMA JOŠ NEKOLIKO PLOTTERA ROLAND DXY 880 A (A3 format) PO POGODNOJ CENI 3 260.000 - dinara



PRVI SUSRET

BASIC-a jednom jedinom naredbom u opsegu od 300 do 2400 bauda. Sprajtovima se barata vrlo jednostavno iz BASIC-a, a efekti su začuđujući. Orao ima i odličan operativni sistem za komunikaciju s najobičnijim kazetofonom, tako da se ovaj doslovno pretvara u jedinicu magnetnih traka. Ideja je slična onoj na kojoj se bazire Commodore-ov dataset (specijalni kazetofon), ali ovdje je u pitanju obični kućni kazetofon. Velika vrlina stare verzije Orla bila je baš u tome što je postignuta vrlo pouzdana komunikacija s kazetofonom. Ova je činjenica nadograđena rutinama za odlaganje PODATAKA na kazetu, tako da je omogućeno kreiranje pravih baza podataka na kazetama.

Orao može poslužiti i kao mali poslovni sistem, što je u praksi i dokazano. Orao opremljen dvostrukom disketnom jedinicom, 80 kolonskom karticom (omogućuje prikaz teksta na ekranu u 80 kolona i 25 redova) i štampačem može poslužiti (skoro) svuda: na kancelarijskom stolu, u liječničkoj ordinaciji...

Odmah nakon uključivanja računala ili reseta javlja se MONITORSKI znak(•) MINIASEMBLER, koji je sastavni dio MONITORA, spreman je za rad. Nije potrebno nikakvo učitavanje sličnih programa s kazete. Pisanje programa u miniasembleru pravo je zadovoljstvo, ali on sam za sebe ne bi bio tako posebno efikasno oruđe kao uz njega ne bi postojao i DISASEMBLER, a zajedno mogu vrlo mnogo. Ostatak monitora su standardne operacije za rad s memorijom, s blokovima memorije, pregled i mijenjanje memorijskih lokacija pojedinačno ili u blokovima.

Uz mikroračunalo Orao PEL iz Varaždina proizvodi i ostale komponente kompletnog mikroračunarskog sistema Orao: video monitor, 80-kolonski i 40-kolonski štampač, disketnu jedinicu i kazetofon.

Što se tiče programske podrške za mikroračunalo Orao, situacija je dugo vremena bila vrlo kritična. Međutim, u posljednjih godinu-dvije dana pojavilo se nekoliko klubova širom Jugoslavije koji polako prerastaju u Yugo-softverske kuće. Njihovim radom do današnjih dana napravljen je priličan broj aplikativnih programa. Najistaknutiji klub koji se bavi izradom programa za Orao je Klub mladih informatičara — Borovo (KMI Borovo), koji je na prijedlog PEL-a postao ekskluzivni isporučilac softvera za mikroračunalo Orao. U svojem katalogu programa KMI Borovo ima oko stotinu komercijalnih programa koji su podijeljeni u četiri skupine: igre, obrazovni programi, uslužni programi i programski jezici. Navest ćemo samo nekoliko naslova. Među igrama ima najviše arkadnih igara: Space invaders, Jumping Jack, Rocky, West bank, Breakout... Najnovije igre su se svojom kvalitetom ne samo približile nego i izjednačile s kvalitetom igara na mnogo popularnijim mikroračunalima (ZX Spectrum, Commodore 64), a to su igre kao Manic Miner, Boulder Dash... Postoji i nekoliko avanture logičkih igara... Među obrazovnim programima postoje programi iz područja fizike, matematike, kemije, geografije, rječnici stranih riječi... Među uslužnim programima ističe se program za obradu teksta TEXTED i program KONVERZIJA koji prevodi programe pisane u staroj verziji BASIC-a u novu proširenu verziju BASIC-a. Među programskim jezicima tu su, uz BASIC i miniasembler, ASSEMBLER, FORTH i mini PASCAL.

Što se literature tiče, postoji nekoliko priručnika i knjižica napisanih za Orla. Uskoro iz štampe izlazi i najnoviji priručnik za programiranje za mikroračunalo Orao u izdanju PEL-a, a u pripremi je i još nekoliko naslova: Mašinski jezik za Orao, Orao-disasemblirani ROM i Orao-DOS.

Orao se već prilično afirmirao kao školsko računalo. Svakim danom u naše (uglavnom) osnovne škole putuje sve više Orlove, a putovat će ih i više kad se do kraja završi projekat o povezivanju Orlova u MREŽU.

Mreža Orlova već radi, rezultati su više nego dobri, a njome je već opremljeno nekoliko probnih kabineta u našim školama. Komunikacija između nastavnika i učenika uspostavlja se prema želji, odnosno prema potrebi. Ograničenja u broju računala koja je moguće povezati u mrežu ne postoje — ona su međusobno povezana običnim trožilnim kablovima, a mogućnostima prijenosa podataka nema kraja. Naime, moguće je prenositi BASIC programe, pojedinačne BASIC naredbe, sadržaje ekrana, varijable i poruke uz fantastičnu brzinu prijenosa od oko 25000 bauda.

Vjerujemo da je sve ovo što je dosad postignuto s mikroračunalom ORAO i 5000 komada prodanih računala širom zemlje pomoglo mnogim učenicima i nastavnicima da naprave svoje prve korake u programiranju. Osnovna namjena mikroračunala Orao upravo je ta: prvi susret s mikroračunalom, učenje programiranja, pisanje prvih programa. Za to je Orao idealno računalo koje svakom početniku može dati sva osnovna znanja i objasniti mikroračunarske [illegible] otvoriti vrata u svijet današnje genera[illegible] bitnih personalnih kompjutera.

6 MUKA SA SOFTVEROM

Obrazovni softver

Dva su viđenja korišćenja računara u obrazovanju — računar kao predmet izučavanja i računar kao nastavno sredstvo. O računarski podržanom učenju još uvek nedovoljno govorimo, našto zbog toga što nemamo dovoljno praktičnih iskustava u ovoj oblasti, a našto i stoga što u našoj sredini još uvek nije shvaćen značaj obrazovnog softvera. Bez obzira na to, računari masovno ulaze u naše škole. Ulaze u sredine koje nisu dovoljno pripremljene da ih prihvate, tako da su ponekad učenici otvoreniji za tu inovaciju od nastavnika. Samo je mali broj nastavnika u vaspitno-obrazovnim organizacijama u Jugoslaviji osposobljen za nastavu informatike, rukovanje računarima i korišćenje računara u nastavi. Istraživanja su pokazala da je zanemarljiv broj nastavnika u našoj zemlji koji se služe računarima u svom radu. Zato možemo reći da je nastupio zadnji čas da se prosvetni radnici, kao i oni koji se spremaju za taj poziv, upoznaju s osnovnim činjenicama o primeni računara u prosveti i pedagogiji.



Zahvaljujući jeftinim mikroračunarima i sve boljem edukativnom softveru u razvijenim zemljama većina škola koristi računare za podršku nastave a samo desetak godina ranije ovo moćno nastavno sredstvo koristilo je svega 10% američkih škola. Pri tom je u prvo vreme računar korišćen prevashodno u tzv. programiranoj nastavi dok se danas sve više koristi za modeliranje i simulaciju.

Programirana nastava

Od svoje pojave u SAD pedesetih godina do danas programirana nastava je doživela značajan uspeh i podstakla razvoj i proizvodnju raznih tehničkih sredstava programiranih udžbenika i softvera. Programirana nastava je vrsta nastave u kojoj se sadržaji na poseban način logički strukturiraju i daju učenicima u manjim ranije pripremljenim delovima. Ove celine oni uče samostalno i postupno idući korak po korak sopstvenim ritmom uz stalno proveravanje stepena usvojenosti izloženih sadržaja.

Računari, kako veliki tajmšering sistemi tako i mikroračunari predstavljaju idealno sredstvo za realizaciju programirane nastave. Postoji više vrsta programirane nastave, a programi koji na ovaj način podržavaju učenje sastoje se obično iz četiri dela. Na početku se nalazi **tekstualni materijal** koji objašnjava o čemu se radi. On mora biti jednostavno i jasno napisan. Zatim se zadaju **tekstualna pitanja** koja zahtevaju odgovor učenika. Računar analizira odgovor. Ako je odgovor tačan dobija se potvrda, a ako nije ukazuje na probleme na koje treba obratiti pažnju. Uvek su predviđeni i **alternativni tekstovi** na istu temu koje obično pišu razni profesori tako da će verovatno svaki učenik naći pristup problemu koji njemu najviše odgovara. Na kraju se vrši **analiza uspeha** koja pokazuje koliko je tema savladana, daje procenat tačnih odgovora i sve druge pokazatelje koji su interesantni za nastavnika.

Međutim, sve češće se čuje da računarski podržana programirana nastava uglavnom pomaže poboljšanju uspeha lošijih učenika omogućavajući im da pristignu svoje vršnjake, ali da ima vrlo malo uticaja na razvijanje kreativnih sposobnosti. Posebno ako se zasniva na biheviorističkoj teoriji koja u središte programirane nastave stavlja odgovor (reakciju) učenika na moguće alternative rešenja i time sputava stvaralačko mišljenje.

Uz to se ne sme izgubiti iz vida da se njenom primenom smanjuje mogućnost za razvijanje socijalnih odnosa i kolektivnih veza što, ako škole ne posmatramo samo kao obrazovne već i kao vaspitne ustanove, predstavlja značajnu slabost svake programirane nastave, pa i one podržane računarima. Međutim, razumno i odmereno korišćenje računara kao sredstva za realizaciju programirane nastave omogućava znatno efektivniji rad. Na ovaj način znatno se poboljšava kvalitet organizacije nastavnog procesa. Što je još bitnije, omogućava se individualno usvajanje znanja sopstvenim ritmom, čime se učenici stimulišu za stalno i aktivno učenje.

Mogućnosti programirane nastave nisu male. One su korisne naročito u matematici, prirodnim i tehničkim naukama. Primena programirane nastave daje dobre rezultate i u učenju gramatike, sintakse i elementarne pismenosti, kao i u mnogim društveno-naučnim predmetima. Kad god su u pitanju saznajne aktivnosti u procesu učenja, programirana nastava može znatno doprineti povećanju efikasnosti, pogotovu ako se u proces učenja uvode problemi koji zahtevaju stvaralačko rešavanje.

No, bez obzira na savršenu tehnologiju i neslućene mogućnosti video diska kao nosioca informacija, ipak programirano učenje neće dati prave rezultate bez prisustva nastavnika. Bez obzira na maštovitost tvoraca softvera, ne mogu se predvideti svi mogući odgovori. Međutim, što se tiče efikasnosti u manipulisanju informacijama, čovek je tu ipak znatno ispod mogućnosti

računara. Zato treba vršiti simbiozu mogućnosti računara i nastavnika i tako na najbolji i najracionalniji način realizovati nastavu.

Učenje otkrivanjem

Modeliranje i simulacija na računaru u nastavi sve više dobijaju na značaju upravo stoga što kod učenika ne podržavaju mehaničko učenje, već podstiču aktivan misaoni rad. Modeliranje u širem smislu reči je postupak zamene jednog objekta drugim, koji mu je u manjoj ili većoj meri sličan. Matematičko modeliranje je zamena opisivanja realnih objekata ili procesa matematičkim relacijama. Ukoliko je ovaj opis kompletniji i tačniji, matematički model je verniji i upotrebljiviji. Matematičko modeliranje je moćno sredstvo u raznim oblastima istraživanja, a koristi se i kao dopuna drugih metoda. U primeni računara u obrazovanju matematičko modeliranje je najpogodniji način „pakovanja" sistema koji se izučavaju u formu koja omogućava upotrebu računara. Jasno je da su neke oblasti više, a neke manje pogodne za ovakav pristup.

Matematički model omogućava pisanje simulacionog programa kojim se prikazuju ponašanje, odnosno osobine modela. Model se posmatra i analizira pomoću računara. Ovakvi simulacioni programi napisani u skladu sa dodatnim zahtevima koje nameće njihovo korišćenje u nastavi, obavezno su praćeni odgovarajućom dokumentacijom. U slučajevima kada treba proučiti složenije, naročito dinamičke sisteme koji imaju komplikovanu strukturu, međuzavisnost sastavnih delova i složeno ponašanje, kad što je to veoma često slučaj u fizici, na primer, pristup preko simulacija predstavlja najpogodniji mogući vid prikaza ovih sadržaja. Simulacionim programima mogu se zamenjivati realni eksperimenti i laboratorijske vežbe koje iz određenih razloga ne mogu da se vrše na realnim sistemima. Razlozi mogu biti cena, dužina trajanja realnih eksperimenata, nepostojanje odgovarajuće opreme i slično.

Šta sve podrazumeva pisanje simulacionih programa možete videti iz posebnog teksta posvećenog ovoj problematici. Mi želimo na kraju još da istaknemo da obrazovni programi, ma kakvog da su tipa, treba da budu otvoreni za moguće izmene i dopune. Njihovim korisnicima, učenicima i nastavnicima, mora se ostaviti mogućnost da uđu u njihovu strukturu i da ih menjaju. Obrazovni programi ne treba da budu zamena već pomoć nastavnicima. Stoga moraju biti propraćeni bogatom i jasnom dokumentacijom. Sem interne dokumentacije koja se bavi specifičnostima samog programa, za edukativne programe treba priložiti i posebnu dokumentaciju koja se obraća nastavnicima, kao i dokumentaciju koja se obraća učenicima.

Pred našim programerima i nastavnicima je ogroman i odgovoran posao kreiranja programske podrške nastave većine predmeta. Ono što već postoji na našem tržištu obrazovnog softvera možete videti u katalozima koji slede.

Učenje računarskom simulacijom

Ribe u ribnjaku

Veljko Spasić

Za Veliku Britaniju kažu da je evropska prestonica kućnog računarstva, kompjuterskih igara i — obrazovnog softvera. Saradnik „Računara" magistar Veljko Spasić, koji je u Engleskoj nekoliko godina radio na razvoju obrazovnog softvera, pripremio je kratak osvrt na kompjuterizaciju engleskih škola i sažeti jednog programa iz bogatog kataloga engleskog obrazovnog softvera. Iako od engleskih programa naše obrazovanje ne može imati direktne koristi — druge mašine i drugi nastavni programi — odlučili smo da pripremimo ovaj prikaz da bismo nastavnicima i programerima omogućili uvid u teme i osnovna ideje koje se primenjuju u obrazovnom softveru.

Obrazovni sistem Velike Britanije najbolje bi se mogao opisati kao decentralizovani nacionalni sistem. Odgovornost za obrazovanje je podeljena između centralne vlade, koja određuje globalnu obrazovnu politiku, i lokalne uprave koja vodi dnevnu politiku. Upravnici škola i koledža, koji se biraju lokalno, odgovorni su za školske programe, dok je nastavnički kadar zadužen za dinamiku savladavanja gradiva, izbor metoda nastave i detaljan koncept lekcija.

Obrazovanje je obavezno od pete do šesnaeste godine. Velika Britanija ima oko devet miliona učenika i oko 400000 školskih nastavnika. Godišnje se utroši približno 14 miliona funti na obrazovanje.

Sistem provere znanja zasniva se na javnim ispitima koji se polažu u šesnaestoj i osamnaestoj godini života. U šesnaestoj godini se polaže takozvani O-nivo pred nezavisnim ispitnim komisijama, a ne pred nastavnicima koji su izvodili nastavu, mada su oni zastupljeni u komitetima ispitnih komisija. Umesto za O-nivo može se polagati sličan ispit koji je nešto više praktično koncipiran.

Budimo malo Englezi

U osamnaestoj godini polaže se za A-nivo ispit se sastoji od najmanje tri predmeta, mada mnogi učenici odabiraju i polažu i više. A-nivo je uslov za upis na univerzitet.

Nastavnici i školski savetnici iniciraju razvoj školskih programa. Oni daju nove ideje, pišu potrebne sadržaje i proveravaju ih najpre u sopstvenim razredima, a zatim i u razredima svojih kolega. Zatim se materijal koji je zadovoljio proveru stavlja na raspolaganje svima preko udruženja nastavnika, specijalizovanih nacionalnih i lokalnih tela, i konačno preko izdavača.

Ovaj model razvoja školskih programa, odnosno obrazovnih sadržaja, ukorenjen je i na njega se oslanja i uvođenje računara u škole.

Vlada Velike Britanije razvija uvođenje i primenu računara u školama preko dva ministarstva: Ministarstva za obrazovanje i nauku i Ministarstva za trgovinu i industriju. Ova ministarstva tesno sarađuju u vezi s ovim pitanjima ali, grubo govoreno, nadležnosti su podeljene. Ministarstvo za obrazovanje i nauku je zaduženo za dodatno obrazovanje nastavnika, za razvoj školskih planova i programa, i za proizvodnju potrebnog softvera i pisanog materijala. Ministarstvo za trgovinu i industriju se stara za obezbeđenje opreme, to jest računara. Vlada je zauzela stav da je proizvodnja dobrog obrazovnog softvera centralno pitanje u čitavoj akciji. Tome se posvećuje najveća pažnja i finansijska podrška.

Najznačajniju ulogu u razvoju obrazovnog softvera ima svakako veliki nacionalni projekat MEP — Microelectronic Education Program — koji je započeo 1980. godine sa velikim godišnjim budžetom od pet miliona funti. Osnovni cilj ovog projekta je da razvije korišćenje računara u obrazovanju.

Da bi ostvario cilj, MEP se oslanja na lokalne obrazovne vlasti i na nacionalne projekte za razvoj obrazovnog softvera. Podržava se usavršavanje nastavnika (kurseve usavršavanja iz raznih oblasti pohađa 28000 nastavnika svake godine, a posebno razvijene pakete za samostalno usavršavanje čak 65000 nastavnika godišnje) kao i razvoj školskih planova i programa.

Zvaničan stav obrazovnih vlasti u Velikoj Britaniji je da svaka škola koja koristi računare prvi put mora dobiti saveta i podršku u sledećim pitanjima:

— sadržaj školskog programa,
— kako uklopiti računar u proces nastave,
— izbor hardvera i softvera,
— izbor najpogodnijih nastavnih metoda koje treba koristiti pri upotrebi računara u obrazovanju,
— izbor pratećeg materijala koji podržava upotrebu računara (knjige, video kasete, itd.),
— upravljanja računarskom učionicom.

Prvi u svetu

Osim ove početne podrške, zadovoljavaju se i ostale potrebe škola: obuka kadrova, održavanje računara, katalоška biblioteka raspoloživog obrazovnog softvera, i u nekim slučajevima saradnja sa iskusnim programerima radi eventualne realizacije ideja nastavnika neračunarskih predmeta u pisanju obrazovnih programa.

Lokalne obrazovne vlasti pomažu realizaciju ovih zadataka naknadom dela troškova nabavke računarske opreme, finansiranje kadra koji obavlja savetodavnu ulogu, i ako je to potrebno, finansiranjem programerskih usluga i usluga velikih računskih centara. Ove vlasti su takođe zadužene za organizaciju i finansiranje dodatnog usavršavanja nastavnika (angažuju se univerzitetski potencijali).

Dugoročna i stabilna vladina politike u domenu primene računara u obrazovanju, kao i odgovarajuća značajna finansijska podrška, predstavljaju osnovu na kojoj se zasniva današnji, možemo slobodno reći, uspešan razvoj ove oblasti u Velikoj Britaniji. Ovome svakako treba dodati i neosporno važan preduslov, a to je postojanje dovoljnog broja kompetentnih stručnjaka koji su u stanju da razvijaju sve složeniju oblast računarski podržanog učenja. Kvalitet i uspešan razvoj osnovne su karakteristike obrazovnih programa koji su široko ušli u nastavu u školama i univerzitetima Velike Britanije i veoma velikog broja drugih zemalja u svetu gde su Britanski obrazovni programi već dugo prisutni i veoma traženi.

Računarska velesila SAD, na primer, takođe intenzivno radi u ovoj oblasti, ali su njihovi obrazovni programi veoma često bez dovoljno inventivnosti i opravdanja za korišćenje računara. Primera radi, jedno nedavno istraživanje kvaliteta obrazovnih programa u Americi, koje su na uzorku od 163 programa vodili Blaio i Erikson 1985, pokazalo je da 65% analiziranih programa ne poseduje jasno utvrđen cilj, bilo obrazovni ili neki drugi; 20% ima jezičke ili pravopisne greške; 60% ne poseduje mogućnost provere napredovanja učenika koji se služe programom (ovo je posebno negativna karakteristika kada se ima u vidu da je preko 70% analiziranih programa zasnovano na metodu uvežbavanja), i konačno zapanjujući podatak da čak 75% programa ne poseduje.

Podrška pri učenju

Primena računara u obrazovanju zajednički je naziv za širok spektar različitih oblasti. Možemo ih podeliti na dve osnovne grupe: prva, računar kao predmet izučavanja u hardverskom i softverskom domenu, i druga, računar kao nezaobilaz

no pomoćno sredstvo odnosno podrška kako pri proučavanju širih informatičkih sadržaja, tako i pri proučavanju i korišćenju računara u brojnim vaninformatičkim disciplinama.

Jedna od oblasti ove druge grupe je primena računara u nastavi različitih predmeta kao što su fizika, biologija i slično. Računar dakle pomaže uspešnijem učenju sadržaja koji ne moraju biti vezani za informatiku ili računarstvo. Ovo ističemo radi toga što se najčešće pod pojmom „računar u obrazovanju" podrazumeva samo izučavanje računara i informatičkih sadržaja.

Upotreba računara kao podrške pri učenju zasniva se na posebno pisanim obrazovnim programima. Ovim tekstom prikazuje se jedan tipičan obrazovni program zasnovan na metodu modeliranja i simulacije. Program o kome je reč publikovan je u Velikoj Britaniji pod imenom BIOMASS (Biomasa). Omogućava simulaciju ekološkog sistema — razvoja populacije riba u ograničenom staništu tokom više godina sa uticajem čoveka ili bez tog uticaja. Program se koristi u nastavi u Velikoj Britaniji i još nekoliko zemalja.

Lična karta

OBLAST Biologija
AUTORI Alison Rose, Veljko Spasić, John Sotillo, Computer in the Curriculum Project, University of London
RAČUNAR BBC, BBC NET, RML 380Z, RML 480Z, RML NET, APPLE
CENA 18.50 funti
IZDAVAČ Longman Micro Software, Longman House, Essex, UK

U tekst prikaza je uključeno ono što prirodno i pripada jednom obrazovnom simulacionom programu: opis sistema koji se proučava, odnosno simulira, razvijeni matematički model koji služi kao osnov za računarsku simulaciju, opis samog simulacionog programa sa grafičkim prikazom rezultata, kratak opis prateće dokumentacije.

Iscrpnost u prikazu ima pored ostalog za cilj da jasno pokaže da pisanje obrazovnih programa nije nimalo jednostavan zadatak. U to je neophodno uložiti dosta napora i znanja, što se u našoj sredini najčešće ne shvata dovoljno ozbiljno.

Osnovni elementi sistema

Slatkovodni ribnjak je ekološki sistem u kome jedinke provode čitav život kao jedina populacija. U njemu su prisutne sve zakonitosti razvoja i međudejstava različitih starosnih grupa unutar populacije, kao i uticaji spoljnih faktora kao što su vremenske prilike, ishrana, izlovljavanje i slično, ali ne postoji kompeticija sa drugim životinjskim vrstama.

Opisaćemo, ukratko, elemente ovog živog sistema, kao i faktore i hipoteze uzete u obzir pri formiranju matematičkog modela.

Posle početnog poribljavanja, to jest punjenja ribnjaka određenim brojem jedinki različitih starosnih grupa (ili samo jednom starosnom grupom) u kojima su zastupljeni i mužjaci i ženke, nastaje razvoj populacije. Svaka jedinka iz ma koje starosne grupe može preživeti u sledeću godinu, pri čemu joj se povećava biomasa, dužina i starosna grupa. Može uginuti zbog starosti ili sa određenom verovatnoćom i ako je mlada, a ženka može položiti ikru ako je polno zrela. Iz položene i oplođene ikre razvijaju se mlade jedinke.

Na razvoj populacije utiču spoljni faktori, kao što su temperatura, ishrana, zagađenost, izlovljavanje, kao i distribucija izlovljenih jedinki po starosnim grupama itd., i unutrašnji faktori, kao

U modelu su [illegible] promenljive i parametri:

$N_t^{(k)}$, k = [illegible] — broj jaja u ikri, broj tek izleglih, broj jednogodišnjih, broj dvogodišnjih, ..., broj osmogodišnjih jedinki u tekućoj godini t_i

$N_{t-1}^{(k)}$ — broj jaja u ikri, broj tek izleglih, broj jednogodišnjih, broj dvogodišnjih, ..., broj osmogodišnjih jedinki u prethodnoj godini t_{i-1}

$N^{(k)}$, k = 1, ..., [illegible] — broj jaja u ikri, broj tek izleglih, broj jednogodišnjih, broj dvogodišnjih, ..., broj osmogodišnjih jedinki izlovljenih u tekućoj godini t_i

n_t — ukupan broj jedinki (zbir brojnosti svih starosnih grupa) u tekućoj godini t_i

n_{t-1} — ukupan broj jedinki (zbir brojnosti svih starosnih grupa) u prethodnoj godini t_{i-1}

$n_t^{(2)}$ — ukupan broj izlovljenih jedinki (zbir brojnosti svih izlovljenih jedinki po starosnim grupama) u tekućoj godini t_i

$B_t^{(k)}$, k = [illegible] — biomasa (u kg) tek izleglih, biomasa jednogodišnjih, biomasa dvogodišnjih, ..., biomasa osmogodišnjih jedinki u tekućoj godini t_i

$B_{t-1}^{(k)}$, k = [illegible] — biomasa (u kg) tek izleglih, biomasa jednogodišnjih, biomasa dvogodišnjih, ..., biomasa osmogodišnjih jedinki u prethodnoj godini t_{i-1}

$B_t^{(k)}$, k = [illegible] — izlovljena biomasa (u kg) tek izleglih, jednogodišnjih, dvogodišnjih, ..., osmogodišnjih jedinki u tekućoj godini t_i

Slika 1 Jednakosti matematičkog modela

$b_t^{(1)}$ — ukupna biomasa (zbir svih biomasa po starosnim grupama) u tekućoj godini t_i

$b_{t-1}^{(1)}$ — ukupna biomasa (zbir svih biomasa po starosnim grupama) u prethodnoj godini t_{i-1}

$b_t^{(2)}$ — ukupna izlovljena biomasa (zbir svih izlovljenih biomasa po starosnim grupama) u tekućoj godini t_i

$w^{(k)}$, k = 1, 2, ..., 9 — srednje vrednosti težina jedinki po starosnim grupama

$r^{(k)}$, k = 1, 2, ..., 9 — srednje vrednosti koeficijenata preživljavanja iz mlađe u višu starosnu grupu

$l^{(k)}$, k = 1, 2, ..., 9 — srednje vrednosti dužine jedinki po starosnim grupama

f — ulov u tekućoj godini t_i, izražen u procentima biomase

l_t — minimalna dužina jedinki koje se izlovljavaju u tekućoj godini t_i

u — slučajna promenljiva sa Gausovom raspodelom

m — maksimalni kapacitet staništa izražen u ukupnoj biomasi (kg)

p — broj jaja u položenoj ikri na svaki kilogram biomase

t — vreme (ne pojavljuje se eksplicitno u modelu), korak jedna godina; t_i — tekuća godina, t_{i-1} — prethodna godina

[illegible] (2)

[illegible] (3)

[illegible] k = 4, ..., 10 (4)

[illegible] k = 4, ..., 10 (5)

jednačina (7), ..., (9), (10) ... k ∈ {[illegible] 1 < k ≤ 9} (6)

[illegible] (7)

$$N^{(k)}_{t} = \frac{B^{(k)}_{t}}{w^{(k)}} \quad (8)$$

$$B^{(k)}_{t} = B^{(k)}_{t}\left(1 - \frac{f_t}{100}\right) \quad (9)$$

$$N^{(k)}_{t} = \frac{B^{(k)}_{t}}{w^{(k)}} \quad (10)$$

$$b_t = \sum_{k}^{10} B^{(k)}_{t} \quad (11)$$

$$n_t = \sum_{k}^{10} N^{(k)}_{t} \quad (12)$$

$$b_t = \sum_{k=2}^{10} B^{(k)}_{t} \quad (13)$$

$$n_t = \sum_{k}^{10} N^{(k)}_{t} \quad (14)$$

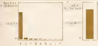
Početni uslovi predstavljaju početnu distribuciju brojnosti po starosnim grupama

$$t = 1 \quad N^{(k)}_{(0)} = N^{(k)}_{(0)} \quad k = 2, 3, \ldots, 10 \quad (15)$$

Slika 2 Promenljive matematičkog modela



Slika 3a Početni uslovi u jednom simuliranom eksperimentu u režimu prikaza histograma brojnosti. U gornjem pravougaoniku je označena tekuća godina (1), procenat izliva u tekućoj godini (0%) i ograničenje na dužinu pri izlovu (0); u levom pravougaoniku na apscisi su redom označene tek izvedene, jednogodišnje, dvogodišnje, ... osmogodišnje jedinke, a na ordinati broj jedinki, u desnom pravougaoniku je predstavljen ukupan broj svih jedinki; početno poribljavanje je postavljeno relativno nepovoljno po razvoj populacije radi toga što nema polno zrelih jedinki. Posle toga može se zahtevati simulacija razvoja populacije. Rezultati se daju za svaku godinu



Slika 3b Posle trideset godina. Populacija je dostigla graničnu stabilnu distribuciju koja se malo menja tokom narednih godina — stanište je zasićeno

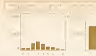
što su verovatnoća preživljavanja iz niže u višu starosnu grupu, koja zavisi od uzrasta i drugih faktora, vreme dostizanja polne zrelosti i polna produktivnost, diferencirani uticaj zasićenja staništa različitim starosnim grupama, kašnjenje prisutno u prenosu spoljnih uticaja kroz sistem, efekat postepenog prigušivanja rasta populacije koji je uzrokovan postojanjem maksimalnog kapaciteta staništa, itd.

Izložićemo model slatkovodnog ribnjaka pastrmki, tako da se pojedine konkretne vrednosti (svi parametri: maksimalna starost, doba polne zrelosti, količina položene ikre itd.) koja figurišu u modelu i simulacionom programu odnose na ovu vrstu.

Početno poribljavanje definiše se jednakostima (15) matematičkog modela prikazanog na slici 2. Ovde se mora voditi računa da ekvivalentna inicijalna biomasa ne prekorači maksimalni kapacitet staništa, a takođe i da svaka uneta starosna grupa smanjuje preostali kapacitet. Ovo je pod kontrolom programa kojim se vrši simulacija modela.

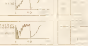
Srednja vrednosti koeficijenata preživljavanja iz niže u višu starosnu grupu zavise od uzrasta jedinki a koriguje ih kriva zasićenja zavisna od maksimalnog kapaciteta staništa, tekuće ukupne biomase i starosne grupe. Najmanje je preživljavanja iz tek izvedenih u jednogodišnja jedinka (kod pastrmki je ova verovatnoća svega oko 0,05, tj. oko 5%); posle dostizanja granične starosti preživljavanja je 0, a tokom životnog veka varira sa uzrastom.

Polna zrelost se dostiže u četvrtoj godini



Slika 3c Histogram biomase (prikaz je analogno organizovan). Distribucija biomase je potpuno različita od distribucije brojnosti po starosnim grupama, granična stabilna distribucija biomase, koja je tipična u živim sistemima, dostiže se i u modelu

Slika 3d Eksperiment sa modelom u kome je jedino mlađ korišćena za inicijalno poribljavanje a izlov je vršen samo u jednoj godini; prikaz je u vidu grafika, što predstavlja alternativni oblik izdavanja rezultata; male tabele uvek daju numerička vrednosti ukupne biomase i broja jedinki samo u tekućoj i prethodnoj godini



života. Radi toga se u jednačini (1) sabiranje vrši od k=6. Količina položene ikre je proporcionalna biomasi polno zrela jedinke. Srednja vrednost koeficijenta koji predstavlja broj jaja u položenoj ikri na svaki kilogram biomase kod pastrmki je 1600 i figuriše u jednačini (1) modela. Kako je biomasa funkcija brojnosti jedinki, prosečnih težina po starosnim grupama i stohastičkog uticaja spoljne sredine, to se ovi faktori održavaju i na količinu položene ikre (jednačine (1), (3) i (5) modela). U modelu je pretpostavljeno da su među polno zrelim jedinkama ženke zastupljene sa 50% (radi toga je p=800). Koliko će se jedinki izvesti iz položene ikre zavisi od broja jaja u ikri, koeficijenta preživljavanja, stepena zasićenosti staništa i ukupne biomase. Jednačina (2) modela daje broj tek izleglih jedinki u tekućoj godini.

Zasićenje staništa

Kako je već pomenuto, stanište ima svoj granični, ili maksimalni, kapacitet. U modelu je to izraženo ukupnom biomasom jedinki. U programu kojim se simulira matematički model maksimalni kapacitet staništa je postavljen na 25000 kilograma.

Biomasa populacije raste u skladu sa svim elementima koji na to utiču, a usporenje ove brzina rasta uzrokovano zasićenjem (odnosno približavanjem ukupne biomase maksimalnom kapacitetu) utoliko je veće ukoliko je stanište popunjenije. Ovaj efekat je prisutan i u modelu. Koeficijenti preživljavanja se koriguju preko krivih zasićenja koja su date u jednačinama (2) i (4) za k=3,4, ..., 10; iz samih jednačina se vidi da je efekat prigušenja funkcija od ukupne biomase, maksimalnog kapaciteta staništa i starosne grupe. Kada ukupna biomasa teži ka maksimalnom kapacitetu, kriva zasićenja teže ka nuli. Na taj način se smanjuju i koeficijenti preživljavanja.

Model garantuje da ukupna biomasa neće preći maksimalni kapacitet. Ali ako bi se u početnom poribljavanju staništa, to jest početnim uslovima (15), greškom dozvolilo prekoračenje ovog uslova, iz modela bi se dobio negativan broj jedinki. Radi toga simulacioni program kontroliše i u dijalogu vodi postavljanja početnih uslova.

Mnoštvo je spoljnih faktora koji deluju na sistem, počevši od vremenskih uticaja kao što su temperatura, padavine, zaleđivanje itd., preko varijabilnosti u količini hrane, pa do uticaja promenljivih koncentracija zagađenja. Veliki broj ovih elemenata, kao i mogućnost njihovog tretiranja kao slučajnih veličina, navodi na uvođenje jedne slučajne promenljive koja bi reprezentovala zbirni spoljni uticaj na sistem. Njeno dejstvo može biti kako pozitivno tako i negativno na brojnost populacija.

Uzeto je da ova slučajna promenljiva ima normalnu raspodelu. Ona u modelu ne deluje direktno na brojnost populacije po starosnim grupama, već utiče na formiranje biomase u tekućoj godini, a na brojnost utiče sa kašnjenjem, kao što je to slučaj i u živom sistemu.

U modelu je ova slučajna promenljiva označena sa w, a na formiranje biomase deluje preko koeficijenata srednjih vrednosti težina po starosnim grupama (u modelu jednačina (3) i (5) za k=3,4, ..., 10).

Simulacioni program u svakom koraku generiše w po algoritmu koji se bazira na ravnomerno raspodeljenoj slučajnoj promenljivoj.

Izlovljavanje pastrmki

Model dozvoljava izlovljavanja jedinki. Za tekuću godinu zadaje se procentualni iznos izlovljane biomase. Izlov može biti različit u svakoj godini ili se godinama može držati stalnim. Ovo se zadaje postavljanjem vrednosti niza f.

Pored toga, u modelu se mogu postaviti ograničenja na minimalnu dužinu jedinki koja se smeju izlovljavati. Ovo se definiše nizom 1, a uslov (6) u modelu obezbeđuje važenje ovog ograničenja. Jednačine (7) — (10) računaju izlovljanu biomasu i broj jedinki po starosnim grupama, kad i preostalu biomasu i preostali broj jedinki po starosnim grupama pri zadovoljenom uslovu (6), radi čega njihov broj varira od minimalno 0 do maksimalno 9. Uključen je i slučaj kada nema izlova.

U modelu se konačno izračunavaju ukupna biomasa i brojnost kako za izlovljane tako i za preostala jedinka u tekućoj godini.

U jednačinama figurišu vrednosti promenljivih u tekućoj i prethodnoj godini. Izračunavanja se vrše sa korakom od jedne godine.

Na slici 1 date su sve promenljiva modela.

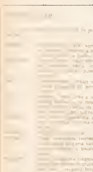
Simulacioni program

Osnovna namena simulacionog programa je da preko interaktivne komunikacije i grafičkih prikaza omogući slobodan eksperiment sa modelom.

Korisnik definiše inicijalnu distribuciju brojnosti jedinki po starosnim grupama, to jest početne uslova (15) iz modela, i režim rada u kome želi izvršavanje programa. To može biti automatska simulacija ili vođena kroz dijalog, sa prikazom rezultata svake godine ili svake desete. U svakoj godini se može postaviti izlov i ograničenje dužine izlovljenih jedinki.

Zatim korisnik zahteva jednu od mogućih vrsta prikaza rezultata simulacije: histogramski prikaz biomase po starosnim grupama i ukupno, ili histogramski prikaz brojnosti jedinki po starosnim grupama i ukupno, ili vremenski grafikon ukupne biomase i ukupne brojnosti jedinki sa ili bez štampane tabela sa rezultatima. U svakom momentu je moguće zameniti bilo koji vid prikaza bilo kojim drugim, i nastaviti od tekuće godine.

Se programom se komunicira preko ključnih reči od kojih svaka izvršava po jednu od tipičnih i potrebnih funkcija. Upotreba ključnih reči je slobodna — mogu se izvršavati ma kojim redosledom. Ovo daje potpunu slobodu u eksperimentu sa modelom



Program analizira unete ključne reči prepoznaje ih i izvršava. Ako korisnik unese nepoznatu ključnu reč ili nedozvoljeni opseg nekog od ulaznih podataka izdaje se poruka iz koje se vidi koja je greška u pitanju i zahteva ponovno unošenje. Time je korisnik obavešten o nastalom problemu i načinu njegovog otklanjanja a program nastavlja normalni red

Ne tipični rezultati simulacije na bazi opisanog modela i programa prikazani su na slici 3

Pri ina je velika oscilatornost broja jedinki nakon inicijalnog poribljavanja. Ovo je direktna posledica namerno loše odabrane početne distribucije jedinki i kašnjenja u sistemu koja uvek izazivaju prigušeno oscilovanje. U realnim sistemima se ova neželjena pojava otklanja pažljivim biranjem početne populacije

Nakon izlova koji je u pedesetoj godini definisan sa istim procentom (90%) biomase u svakoj starosnoj grupi populacije se oporavlja. Ovoga puta sa znatno manjom oscilatornošću (smanjenje je uzrokovano boljom distribucijom) kroz sasvim jasno izraženu s krivu ponovo se dostiže maksimalna brojnost odnosno biomasa u datom staništu

Struktura programa

Program je pisan u bejziku (za to postoje posebni razlozi u koje ovde ne ulazimo) a zaprema oko 30 K memorije. Struktura mu je modularna realizovana pomoću potprograma. Osnovni su dekoder ključnih reči modul za rešavanje jednačina matematičkog modela modul za komunikaciju sa korisnikom nekoliko modula za grafički prikaz po jedan nadgrađeni modul za izvršavanje svake od ključnih reči i veliki broj malih pomoćnih potprograma

Prateća dokumentacija se sastoji od dokumentacije za nastavnike dokumentacije za učenike i tehničke (programske) dokumentacije

Dokumentacija za nastavnike opisuje realan sistem program i njegovu upotrebu. Daje se i spisak dopunske literature

Dokumentacija za učenika sadrži četiri dela 1 Model je realan sistem 2 Korišćenje programa, 3 istraživanja, 4 Lista ključnih reči. Primera radi u delu 3 učenici se usmeravaju da traže odgovore na pitanja kao što su

— Koliko treba populaciji da se stabilizuje i zašto dolazi do stabilizacije?

— Opišite i objasnite fluktuacije u rastu populacije

— Uporedite produktivnost posle poribljavanja starosnim grupama 4—8 sa produktivnošću posle poribljavanja samo sa mladi

— Koje je najpogodnije vreme za izlov tokom rasta populacije sa ciljem da se ostvari maksimalni održivi prinos?

Programska dokumentacija je veoma obimna i sadrži između ostalog detalje o svim potprogramima svim promenljivima i svim grafičkim prikazima koji se javljaju u programu. Iscrpnost u dokumentaciji je neophodna radi toga što se mora omogućiti eventualno usavršavanje programa a i obezbediti da nastavnici (ili učenici) ako to žele modifikuju elemente u programu modelu grafičkim prikazima i slično

Sve tri dokumentacije su obima približno 20+10+100 kucanih strana

Umesto zaključka

U oblasti primene računara u obrazovanju moguće je koristiti i razvijati različite metode a svaki će sigurno imati i dobrih i loših strana. Ovde opisan program se oslanja na modeliranje i simulaciju, dakle na metod koji u velikoj meri omogućava slobodan eksperiment pri učenju. To je metod koji je u ekspanziji jer zaista koristi potencijale računara a nije samo komplikovanija (a time i lošija) zamena za knjigu ili školsku tablu — što postupak koji je prilično raširen u primeni računara u obrazovanju

Izvesno je međutim da računar još nije na dovoljno osmišljen i efikasan način iskorišćen u ovoj oblasti. Godina koja su pred nama verovatno će doneti značajnije prodore

„Računari" i dalje pišu za škole

Pored niza aplikativnih i programerskih tema „Računari" u svakom broju objavljuju i poseban blok o primeni računara u obrazovanju pod naslovom „Pet plus" sa testima iz škole metodičkim uputstvima za izvođenje nastave i primerima iz pedagoške prakse

Katalog Yu softvera

Ninoslav Čabrić

O jugoslovenskom tržištu softvera najčešće se govori s krajnjim pesimizmom — ono je, rekło bi se, sasečeno u samom korenu. U jednoj oblasti, međutim, nije sve tako crno. Uprkos svemu što ih ometa, ipak postoje ljudi koji pišu obrazovni softver i kuće koje su spremne da ga objave. Možda će neko primetiti da se radi o jednostavnim programima za zastarele mašine, ali se ne može poreći da katalog od šezdesetak obrazovnih programa ipak predstavlja zametak domaće industrije softvera.

Katalog je načinjen na osnovu uvida u jugoslovensku produkciju obrazovnih programa za računare. Navedeni su programi koji se mogu kupiti u slobodnoj prodaji ili kod izdavača a u postupku izdavanja ocenila ih je recenzentska komisija. Katalog sadrži 67 programa grupisanih u nastavne oblasti. Kako se na nekim kasetama nalazi više programa koji predstavljaju logičku celinu oni su prikazani kao jedan program. Takođe nisu posebno navođeni programi koji predstavljaju verziju već prikazanog programa napisanu za računar drugog tipa

Mada je učinjeno sve da se ni jedan postojeći program ne izostavi unapred se izvinjavamo autorima i izdavačima čiji programi omaškom nisu ušli u ovaj prikaz. Molimo da o programima koji nisu obuhvaćeni obavestite redakciju

Astronomija

PLANETI

ASTRONOMIJA, („spektrum 48K)
Bojan Dintinjana i Andrej Čadež
Državna Založba Slovenije. Zavod SR Slovenije za školstvo (Ljubljana 1986)

Program omogućuje simulaciju kretanja planeta oko Sunca i daje njihove položaje za posmatrača na Zemlji. Osim grafičkih prikaza date su i tabele sa osnovnim astronomskim podacima neophodnim za posmatranje planeta na nebeskoj sferi

PLANETE

ASTRONOMIJA 1, („spektrum 48K)
Ninoslav ČABRIĆ i Sava JEREMIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB Beograd (II izdanje 1986)

Ovaj program namenjen je proračunavanju položaja Sunca Meseca i planeta na nebeskom svodu. Moguće je zadati koordinate mesta na Zemlji sa koga se posmatra nebeski svod. Pored toga položaji se dobijaju za zadato vreme. Programom se može pripremiti posmatranje planeta na nebeskom svodu i proučiti kako se planete prividno kreću

JUPITEROVI SATELITI

ASTRONOMIJA 1, („spektrum 48K)
Ninoslav ČABRIĆ i Sava JEREMIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB Beograd (II izdanje 1986)

Program simulira kretanje Jupiterovih satelita Io Evrope Ganimeda i Kalista oko matične planete. Prikazivanje satelita je dvojako u dnu ekrana data je slika kako sateliti izgledaju kada se gledaju kroz teleskop sa Zemlje a u centralnom delu ekrana je slika kako bi se Jupiterov sistem video gledan visoko iznad ravni Sunčevog sistema.

HALEJEVA KOMETA

ASTRONOMIJA 1 („spektrum" 48K)
Ninoslav Čabrić i Sava Jeremić
Zavod za udžbenike i nastavna sredstva i PGP RTB
Beograd (drugo izdanje 1986)

Program omogućuje pronalaženje Halejeve komete na nebeskom svodu. Dat je položaj komete na putanji oko Sunca i na nebeskoj sferi posmatrano sa proizvoljnog mesta na Zemlji i u izabranom trenutku.

SUNCE I MESEC

ASTRONOMIJA 1 („spektrum" 48K)
Ninoslav Čabrić i Sava Jeremić
Zavod za udžbenike i nastavna sredstva i PGP RTB
Beograd (drugo izdanje 1986)

Ovaj program nudi efemeridske podatke za Sunce i Mesec. Namenjen je pripremi astronomskih posmatranja i boljem shvatanju procesa koji se oko nas dešavaju. U program su inicijalno uneti podaci za Beograd, ali se mogu jednostavno promeniti za ma koje drugo mesto na Zemljinoj kugli.

OSVAJANJE PLANETA

OSVAJANJE PLANETA („spektrum" 48K)
Boris MALINAR i Goran BARTOLIĆ
Filmoteka 16, Zagreb (1985)

Cilj programa je „poseta" svim planetama Sunčevog sistema. Da bi se to postiglo, potrebno je lansirati raketu, uvesti je u orbitu oko Zemlje, usmeriti je prema odabranoj planeti, ući u orbitu oko nje, meko spustiti automatsku sondu i pomoću nje istražiti planetu, zatim se usmeriti nazad ka Zemlji, ući u orbitu oko nje i uspešno se prizemljiti. To nije sve. Kraj jedne misije je kada se preda izveštaj specijalnoj Komisiji i kada ga ona usvoji (tu se moraju izneti svi podaci o planeti koja je posećena). Program je namenjen učenicima počev od V razreda osnovne škole.

SATELITI 2

ASTRONOMIJA („spektrum" 48K)
Bojan Dintinjana i Janez Jaklič
Državna Založba Slovenije, Zavod SR Slovenije za školstvo (Ljubljana 1986)

Program prikazuje let veštačkih satelita čiji se parametri kretanja mogu zadati. Dat je grafički prikaz projekcije kretanja satelita na površinu Zemlje. Prikazani su razni parametri kretanja.

SATELITI 3

ASTRONOMIJA („spektrum" 48K)
Bojan Dintinjana i Janez Jaklič
Državna Založba Slovenije, Zavod SR Slovenije za školstvo (Ljubljana 1986)

Program je namenjen za samostalni rad učenika (priprema posmatranja astronomskih pojava i izrada seminarskih radova) ili profesorima za časove obrađivanje ili usvajanje novog gradiva. Najpre se prikazuju najsjajnije zvezde i sazvežđa na željenom delu neba (onako kako bi se videlo kroz foto-aparat čiji je vidni ugao oko 60 stepeni). Moguće je pomeranje, a nebeska koordinate centra vidnog polja slike zapisana su u vrhu ekrana. Pored toga program „ume" da postavlje pitanja o najsjajnijim sazvežđima, daje sve bitne podatke o njima i omogućuje da vidimo kako se menja izgled neba zbog rotacije Zemlje ili zbog pomeranja posmatrača po njenoj površini.

Engleski jezik

FLYING (LETENJE)

ENGLESKI JEZIK 1, („spektrum" 48K)
John HIGGINS
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (II izdanje 1986)

Jednostavan program namenjen potpunim početnicima i u engleskom jeziku i u korišćenju računara. Cilj je učenje i vežbanje značenja i upotrebe engleskih predloga FROM i TO.

JOHN AND MARY (DŽON I MERI)

ENGLESKI JEZIK 1, („spektrum" 48K)
John HIGGINS
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (II izdanje 1986)

Ovaj program je namenjen onima koji su tek počeli da uče engleski jezik. Njegovim korišćenjem uči se korišćenje zapovednog načina.

PHOTOFIT (PORTRET PROVALNIKA)

ENGLESKI JEZIK 1 („spektrum" 48K)
John HIGGINS
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (II izdanje 1986)

Radi se o foto-robotu pomoću koga treba rekonstruisati lik nekog razbojnika. Kao svaki fotorobot i ovaj radi tako što mu se zada koji deo lica treba da crta, a kada to učini, onda se pridevima koriguje nacrtano. Programom PHOTOFIT omogućuje učenje engleskih naziva delova lica i karakterističnih prideva koji ih opisuju.

MURDER (PRONAĐI UBICU)

ENGLESKI JEZIK 1 („spektrum" 48K)
John Higgins
Zavod za udžbenike i nastavna sredstva i PGP RTB (drugo izdanje, Beograd 1986)

U stilu Agate Kristi korisnik dolazi u situaciju da pronađe ubicu postavljanjem pitanja osumnjičenima. U suštini, vežba se formiranje upitnog oblika rečenice.

TEXT SUFFLER (PRODUŽI PRIČU)

ENGLESKI JEZIK 1 („spektrum" 48K)
John Higgins
Zavod za udžbenike i nastavna sredstva i PGP RTB (drugo izdanje, Beograd 1986)

Po jednoj zadatoj rečenici i tri alternativna nastavka korisnik se odlučuje za ispravan nastavak započete priče.

STORYBOARD (BRISANA PRIČA)

ENGLESKI JEZIK 1 („spektrum" 48K)
John Higgins
Zavod za udžbenike i nastavna sredstva i PGP RTB (drugo izdanje, Beograd 1986)

Program predstavlja računarsku verziju poznatog vežbanja: nepoznati tekst se pročita, zatim skloni, a umesto njega dobija tekst zamenjen crticama koje označavaju slova. Osim toga, ostaju samo naznačene praznine između reči i interpunkcijski znaci. Zadatak je da se što pre rekonstruiše što više zadatog teksta. Korisniku je na raspolaganju sedam tekstova različite težine i dužine.

PRINTER S DEVIL

ENGLESKI 1, („spektrum" 48K)
John Higgins
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (II izdanje 1986)

Ovaj program omogućuje vežbanje namenjeno onima koji su osnove engleskog jezika već usvojili i žele da usavrše pisanje engleskih reči. Pomoću računara realizovano je vežbanje u kome „vragolasti štampač" premeće slova u rečima, ubacuje slogove, „guta slova" i na sličan način greši u pisanju engleskih reči. Cilj vežbanja je da se pronađe pravilo po kome u svakom pojedinačnom slučaju „štampač" greši i da se na zadatom primeru (ispravno napisane reči) dokaže da je pravilo nađeno. Ima tri nivoa težine reči (jednostavne, obične i veoma teške reči) i tri nivoa komplikovanosti pravila po kome štampač kvari reči.

CATCH THE VERB

ENGLESKI („orik nova" 64)
ŠKD FORUM, Ljubljana (1986)

UHVATI GLAGOL je prevod naslova programa koji predstavlja računarsku vežbu za nepravilne glagole. U programu su zapisana 135 glagola engleskog jezika. Zadatak je ispravno odrediti koji su od njih pravilni, a koji nepravilni.

PROSTE REČENICE

ENGLESKI JEZIK 2 („spektrum" 48K)
Vladimir ŽEGARAC i Đorđe SENIČIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Programom se uvežbava formiranje jednostavnih potvrdnih, odrečnih i upitnih rečenica.

UPOZNAVANJE

ENGLESKI JEZIK 2 („spektrum" 48K)
Vladimir ŽEGARAC i Đorđe SENIČIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Ovaj program kroz upoznavanje sa računarskim likovima omogućuje učenje i uvežbavanje uobičajenih fraza i konstrukcija koje se u engleskom jeziku upotrebljavaju u upoznavanju sa nekim ili prilikom raspitivanja o nekom.

PREDLOZI

ENGLESKI JEZIK 2 („spektrum" 48K)
Vladimir ŽEGARAC i Đorđe SENIČIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Efektnim programerskim rešenjima uz korišćenje grafike omogućeno vežbanje engleskih predloga.

SLOŽENE REČENICE

ENGLESKI JEZIK 2 („spektrum" 48K)
Vladimir ŽEGARAC i Đorđe SENIČIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

U ovom programu data je mogućnost uvežbavanja formiranja složenih rečenica sa AND, BECOUSE, WHO i WHICH.

REČI

ENGLESKI JEZIK 3 („spektrum" 48K)
Vladimir ŽEGARAC i Đorđe SENIČIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Program omogućuje vežbanje u pisanju pojedinih engleskih reči.

GLAGOLI

ENGLESKI JEZIK 3 („spektrum" 48K)
Vladimir ŽEGARAC i Đorđe SENIČIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Program je namenjen učenju oblika engleskih glagola, posebno nepravilnih.

EDITOR

ENGLESKI JEZIK 3, („spektrum" 48K)
Vladimir ŽEGARAC i Đorđe SENIČIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Program omogućuje pravljenje zadataka (od strane nastavnika) i njihovo rešavanje (od strane učenika). Nastavnik može (uz pomoć računara) da postavi zadatke ili nova vežbanja. Uz program se dobijaju tri teksta koje su autori pripremili za demonstraciju mogućnosti programa. Rešavanje postavljenih zadataka sastoji se u upisivanju reči koje nedostaju ili prevođenju glagola iz infinitiva u odgovarajuće vreme ili ma šta drugo što je zamišljeno u toku postavljanja zadatka.

POGODBENE REČENICE

ENGLESKI JEZIK 3 („spektrum" 48K)
Vladimir ŽEGARAC i Đorđe SENIČIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Uvežbavanje i učenje formiranja pogodbenih rečenica.

ENGLESKO-SLOVENAČKI REČNIK

ENGLESKI JEZIK („spektrum" 48K)
JAKOPIN, KANIĆ i KRAMBERGER
ZOTKS, Ljubljana (1986)

Rečnik sadrži 3175 reči (1112 engleskih i 2063 slovenačkih). Po pravilu svakoj engleskoj reči odgovara ju po dva slovenačke. Prvenstveno je namenjen prevođenju engleskih reči na slovenački, a obrnuto se može postići uz dosta napora.

Fizika

EKVIVALENTNI OTPORI

FIZIKA 1 („galaksija" 6—6)
Milan Radojčić
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Program je namenjen učenicima VIII razreda osnovne škole i može da služi za učenje i uvežbavanje računanja vrednosti ekvivalentnog otpora dobijenog od kombinacije redne ili paralelne veze otpornika. Sadrži 5 mogućih načina vezivanja dva i tri otpora. Potrebno je naći ukupan otpor kola, ukupnu jačinu struje koja protiče kroz kolo, kao i jačinu struje koja protiče kroz pojedine otpornike.

PRORAČUN TRANSFORMATORA

FIZIKA 1, („galaksija" 6—6)
Milan RADOJČIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Program je namenjen onim učenicima (i nastavnicima) koji su zainteresovani za samogradnju raznih električnih ili elektronskih uređaja i potreban im je proračun transformatora. Program omogućuje da se dobije transformator striktno određenih karakteristika sa postojećim materijalom, određenom debljinom žice, određenim tipom jezgra itd.

ZAKONI ELEKTRIČNE STRUJE

FIZIKA 1, („galaksija" 8—6)
Milan Radejčić
Zavod za udžbenike i nastavna sredstva i PGP RTB Beograd (1986)

Program je napravljen tako da računar postavlja zadatke, pomaže kada je potrebno, evidentira broj tačnih odgovora i služi kao kalkulator. Za rešavanje postoji šest osnovnih tipova zadataka sa 4 do 6 pitanja. Pitanja se odnose na zakone električne struje: Omov zakon, Kirhofljeva pravila i Džulov zakon.

VITSTONOV MOST

FIZIKA 1 („galaksija" 8—6)
Milan Radejčić
Zavod za udžbenike i nastavna sredstva i PGP RTB Beograd (1986)

Programom se simulira jedan od klasičnih fizičkih eksperimenata koji omogućuju merenje električnog otpora. Namenjen je učenicima VIII razreda osnovne škole i idealan je za uvežbavanje toka eksperimenata i obrade rezultata merenja.

ZAKON ELEKTRIČNOG OTPORA

FIZIKA 1 („galaksija" 8—6)
Milan Radejčić
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Ovim programom mogućno je uvežbati i naučiti zakon električnog otpora. Program je namenjen individualnom i grupnom radu učenika VIII razreda osnovne škole.

PRELAMANJE SVETLOSTI

FIZIKA 1 („galaksija" 8—6)
Milan Radejčić
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Ovaj program namenjen je učenicima VIII razreda osnovne škole. Osnovna namena programa je da pomogne nastavniku u toku predavanja o prolasku svetlosti kroz prizmu i planparalelnu ploču, kao i o ponašanju svetlosti na granici dva sredine.

URI

URI („spektrum" 48K)
Damen i Janez Ferbar i Miloš Pelcar
Državna založba Slovenije, Ljubljana (1985)

Program URI omogućuje učenje Omovog zakona, Kirhofljevih pravila i nalaženje ekvivalentnih otpora pomoću računara. Posebno je zanimljiva kombinacija otpornika i sijalica kod kojih se otpor menja u zavisnosti od napona na koji su priključeni. Osim računskih zadataka koje treba rešiti, ima dosta zadataka koji se rešavaju kvalitativno: treba odgovoriti da li neka sijalica svetli ili ne i slično. Program je namenjen učenicima VIII razreda osnovne škole, a napisan je na slovenačkom jeziku.

DINAMIC 48

DINAMIC („spektrum" 48K)
Janez Jeklič
Državna založba Slovenije, Ljubljana (1986)

DINAMIC 48, ili DINAMIČKA SIMULACIJA, kako je potpuno ime programa, predstavlja računarsku simulaciju ravanskog kretanja materijalne tačke pod dejstvom proizvoljne sile. Tako se programom, u zavisnosti od izabranih početnih uslova, mogu simulirati slobodno i prinudno oscilatorno kretanje (sa i bez prigušenja), kretanje u polju gravitacione sile (sa otporom vazduha ili bez njega), kretanje u statičnom električnom polju i homogenom električnom ili magnetnom polju. Simulacija kretanja može se predstaviti na dva načina: ispisivanjem numeričkih vrednosti parametara kretanja ili crtanjem trajektorije.

KRETANJE MOLEKULA U GASOVIMA

FIZIKA 1, („spektrum" 48K)
Ninoslav ČABRIĆ i Sava JEREMIĆ
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (II izdanje 1986)

Ovaj program omogućuje da se zada srednji slobodan put koji će molekul prolaziti bez sudara sa drugim molekulima i intenzitet spoljne sile koja eventualno deluje na molekule gasa. Posle zadavanja ovih podataka na ekranu se crta putanja jednog molekula. Izlomljena linija koja nastaje pokazuje kako se menja pravac kretanja molekula posle sudara sa drugim molekulima i zidovima suda.

FOTOEFEKAT

FIZIKA 1 („spektrum" 48K)
Ninoslav Čabrić i Sava Jeremić
Zavod za udžbenike i nastavna sredstva i PGP RTB Beograd (II izdanje 1986)

Program pruža mogućnost za demonstraciju i upoznavanje sa uslovima za nastajanje i osobinama fotoelektričnog efekta. Namenjen je prvenstveno nastavi u II razredu usmerenog obrazovanja. Parametri se unose, a u zavisnosti od toga dobijaju različite slike, dijagrami i simulacije toka procesa. Program je prvenstveno namenjen nastavnicima i profesorima fizike za časove obnavljenje ili izlaganja novog gradiva.

KARNOV CIKLUS

FIZIKA 1, („spektrum" 48K)
Ninoslav Čabrić i Sava Jeremić
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (II izdanje 1986)

Reč je o programu koji obrađuje jednu od metodskih jedinica koja se uči i u osnovnoj i u srednjoj školi a i na mnogim fakultetima — o toplotnim mašinama. Tačnije, obrađena je idealna toplotna mašina u kojoj se određena količina idealnog gasa nalazi u cilindru sa klipom i naizmenično širi i sabija. Program je namenjen nastavnicima i profesorima fizike.

SLAGANJE TALASA

FIZIKA 1, („spektrum" 48K)
Ninoslav Čabrić i Sava Jeremić
Zavod za udžbenike i nastavna sredstva i PGP RTB Beograd (drugo izdanje 1986)

Program omogućuje analizu pojave slaganja dva talasa proizvoljno zadatih amplituda, frekvencija i početnih faza. Namenjen je grupnom i individualnom radu.

KINEMATIKA

FIZIKA 1 („spektrum" 48K)
Ninoslav Čabrić i Sava Jeremić
Zavod za udžbenike i nastavna sredstva i PGP RTB Beograd (drugo izdanje 1986)

Program omogućuje analizu kretanja materijalne tačke i polju gravitacione sile i/ili pod dejstvom početnog impulsa. Istovremeno se crtaju grafici pređenog puta u vremenu, promene brzina u vremenu i trajektorija.

FIZIKA 1

FIZIKA, („komodor" 64)
D. Maďarec i T. Obreht
Rednički univerzitet Maribor

U programu su obuhvaćene sledeće metodske jedinice: zakoni odbijanja i prelamanja svetlosti, ravnomerno kretanje, električni otpor u tečnostima, metalima i vakuumu.

ZRCALO

FIZIKA („spektrum" 48K)
Bojan Dintinjana i Andrej Čadež
Državna založba Slovenije, Ljubljana

Program je namenjen upoznavanju, uvežbavanju i proveri znanja. Obrađena su ravna, sferna, parabolična, eliptična i hiperbolična ogledala u svetlu geometrijske optike.

SATELITI 1

FIZIKA („spektrum" 48K)
Bojan Dintinjana i Janez Jeklič
Državna Založba Slovenije, Zavod SR Slovenije za školstvo (Ljubljana 1986)

Program prikazuje kretanje materijalne tačke u polju gravitacione sile. Simulacija je zasnovana na rešavanju jednačina kretanja u polju gravitacione sile i grafičkom prikazivanju rezultata.

Geografija

ZEMLJOPIS (EVROPA)

ZEMLJOPIS, („spektrum" 48K)
Gorazd Oirešnik
Revija BIT i ZOTEKS, Ljubljana (1985)

Program EVROPA namenjen je učenicima šestog razreda osnovne škole i predstavlja test znanja uz stvaralačko korišćenje grafičkih mogućnosti „spektruma". Može se izabrati jedan od sledećih predmeta testiranja: evropsko Sredozemlje, Podunavlje, srednja Evropa, zapadna Evropa, severna Evropa, politička podela evropskih zemalja. Pored testiranja, program pruža i izvesne mogućnosti za samostalno učenje. Program je napisan na slovenačkom jeziku.

SLIVOVI

HIDROENERGETSKE OSNOVE SFRJ („spektrum" 48K)
Saša Stiković, Bibijana Mihevc i Tatjana Ogrias
Institut za geografiju Univerziteta E. Kardelj i ZOTEKS, Ljubljana (1985)

U programu se najpre obrađena materija kratko izlaže u slici i reči, a zatim se prelazi na testiranje. Pitanje se sastoje u tome da se pogodi koja je reka nacrtana i kom slivu pripada. Obuhvaćeni su podaci o 32 reke iz svih krajeva naše zemlje, koje pripadaju jadranskom, crnomorskom i egejskom slivu.

ORIJENTACIJA KOMPASOM

PRIRODA I DRUŠTVO („orao" 102, 32K)
Miljenko Krajnović
Filmoteka 16, Zagreb

Za učenje i uvežbavanje orijentacije pomoću kompasa učenik ima na raspolaganju plan, kompas i trodimenzionalnu sliku šume u kojoj treba da pronađe četiri objekta. Program je namenjen učenicima III i IV razreda osnovne škole.

ZAKLADI SLOVENIJE

PRIRODA I DRUŠTVO („spektrum" 48K)
Matej i Erik Kuzmi
Mladenske knjige, Ljubljana

Program je napravljen tako da računar vodi korisnika po Sloveniji. Cilj je proći kroz Sloveniju i doći do Slovenačkog primorja.

Matematika

SABIRANJE

MATEMATIKA I TO, („galaksija" 8—6)
Željko Kralić
Zavod za udžbenike i nastavna sredstva i PGP RTB (drugo izdanje, Beograd 1986)

Radi se o jednostavnom programu koji učenike prvog ili drugog razreda osnovne škole trenira da brzo i tačno sabiraju. Na istoj kaseti se nalazi još tri programa koji na sličan način pomažu u usavršavanju oduzimanja, množenja i deljenja.

MNOŽENJE

MATEMATIKA 1, („spektrum" 48K, „galaksija" 8—6, „comodor" 64)
Dušan Bošnjak
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (II izdanje 1986)

Ovaj program omogućuje učenicima II razreda osnovne škole da pomoću računara uvežbaju množenje dva broja. Programom su obrađene sledeće metodske jedinice: tablica množenja, prioritet operacija, množenje zbira i razlike i množenje dvocifrenog broja. Pored toga, mogu se raditi još i problemski zadaci ili zadaci po slučajnom izboru računara. U okviru svake metodske jedinice može se izabrati jedan od sledećih režima rada: izlaganje, uvežbavanje, ocenjivanje i rešavanje. U toku uvežbavanja računar nudi mogućnost da se zadatak rešava postupno ili da se odmah upiše rezultat.

DELJENJE

MATEMATIKA 2, („spektrum" 48K, „galaksija" 8—6)
Dušan Bošnjak
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (II izdanje, 1986)

Program je namenjen učenicima II razreda osnovne škole i predviđen je za individualni rad učenika. Obrađena su sledeće metodske jedinice: osnovni pojmovi i definicije, deljenje sa ostatkom, deljenje kada je rezultat veći od 10 i problemski zadaci. U okviru svake metodske jedinice može se izabrati jedan od sledećih režima rada: izlaganje, uvežbavanje, ocenjivanje i rešavanje.

MANJE-VIŠE

MATEMATIKA 3 („spektrum" 48K)
Žarko Vukosavljević i Aleksa Pajvančić
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (II izdanje 1986)



Program je namenjen učenicima I razreda osnovne škole. Na zanimljiv način najmlađi se upoznaju sa pojmovima manjeg i većeg broja i razvijaju logiku refleksa i koordinaciju.

ABAKUS

MATEMATIKA 3 (spektrum 48K)
Žarka Vukosavljević i Pređrag Milićević
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (drugo izdanje 1986)

Program omogućuje uvežbavanje brzog sabiranja oduzimanja množenja i deljenja i namenjen je učenicima I razreda osnovne škola (u drugom polugođu) i II razreda (na početku godine). Abakus predstavlja dobar primer trening programa na računaru.

JEDNAČINE

MATEMATIKA 4 („spektrum 48K)
Dušan i Marija Bašnjak
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Program je namenjen učenicima II razreda osnovne škole i obrađene su sledeće oblasti: jednostavnije jednačine, složenije jednačine, problemi složeniji problemi i rešavanja jednačina. U svakoj od oblasti učenik dobija zadatak koji može da rešava sam ili da zatraži pomoć od računara. Računar priskače u pomoć i kada zadatak nije tačno urađen, a pogotovu je interesantna opcija u kojoj se zadaju zadaci koje računar rešava uz komentare i objašnjenja.

RAZLOMCI

MATEMATIKA 5 (spektrum 48K)
Dušan Bašnjak i Vladimir Buturović
Zavod za udžbenike i nastavna sredstva i PGP RTB Beograd (1986)

U ovom programu razlomci su obrađeni u šest osnovnih celina programa: uvođenje pojma razlomka, izražavanje celine razlomka, njihovi različiti zapisi, određivanje dela broja, upoređivanje razlomaka i mnoštvo zadataka za vežbanje. Svi pojmovi obrađeni su u skladu sa nastavnim planom i programom za II razred osnovne škole.

FUNKCIJE

FUNKCIJE, („spektrum 48K)
Vladimir Batagelj
Državna Založba Slovenije, Zavod SR Slovenije za školstvo (Ljubljana 1986)

Program omogućuje crtanje grafika i analizira funkcije realne promenljive. Namenjen je učenicima svih razreda srednjih škola kao i studentima prve godine fakulteta na kojima se izučava matematička analiza. Mogu da ga koriste i profesori za pripremu ili na časovima obrađivanja novog gradiva (ili utvrđivanja) ali glavne prednosti pokazuje kao način za brzu proveru urađenih zadataka.

PRVI KORACI U MATEMATICI

MATEMATIKA, („spektrum 48K)
Boris Pilvelić
Filmoteka 16, Zagreb

Program je namenjen učenicima za uvežbavanje aritmetičkih operacija. Može se koristiti samostalno ili u nastavnom procesu.

MACEK MURI ŠTEJE IN RAČUNA

MATEMATIKA („spektrum 48K)
Ljubo Kastrevc
Brut Film Ljubljana

Program je namenjen deci od 4 do 8 godina. Kroz igru dete se uvodi u svet brojeva i osnovnih aritmetičkih operacija.

CICIBAN ŠTEJE (CICIBAN RAČUNA)

CICIBAN (spektrum 48K)
Davor Banešić
ZOTKS Ljubljana

Program namenjen deci predškolskog uzrasta. Dete sabira predmete koji se pritiskom na neki taster pojavljuju na ekranu. Pored slovenačke postoji i srpskohrvatska verzija programa.

DOBER DAN, MATEMATIKA

MATEMATIKA (spektrum 48K)
Jože Nemec
Državna Založba Slovenije, Zavod SR Slovenije za školstvo (Ljubljana 1986)

Program je namenjen učenicima II razreda osnovne škole. Može se koristiti za samostalno učenje ili uvežbavanje osnovnih računskih operacija.

Tehničko obrazovanje

[illegible]

TEHNIČKO OBRAZOVANJE 1 (spektrum 48K)
Nenad Baločanin
Zavod za udžbenike i nastavna sredstva Beograd (1985)

Program je namenjen izvođenju slobodnih aktivnosti u okviru predmeta TEHNIČKO OBRAZOVANJE i članovima klubova vazduhoplovnog modelarstva. Osnovna namena ovog programa je dobijanje proračuna modela aviona po zadatim parametrima. Za zadate ulazne podatke dobija se aerodinamički proračun modela i prikazuju njegove karakteristike uključujući i geometrijske veličine neophodne za praktičnu primenu dobijenih rezultata.

TEST IZ TEHNIČKOG OBRAZOVANJA

MATEMATIKA I TO, (galaksija 6—8)
Željko Krstić
Zavod za udžbenike i nastavna sredstva i PGP RTB Beograd (II izdanje 1986)

Ovaj program namenjen je članovima klubova Narodne tehnike i učenicima za pripremu teorijskog dela na takmičenjima u organizaciji klubova mladih tehničara. Test se sastoji od 20 pitanja. Tačni odgovori se biraju među ponuđenima. Sastavni deo programa je i rang lista u kojoj se daje redosled postignutih rezultata.

ZAŠTITA OD POŽARA,

(spektrum 48K)
Branka i Nenad Žezlina
Filmoteka 16, Zagreb (1985)

U programu je uspešno povezana obrazovna funkcija računara i povećanje motivacije učenika za učenje kroz igru. Reč je o vatrogascu Florijanu kome treba pomoći da ugasi iznenadno nastali požar u kući, garaži ili dvorištu. Za ovo je potrebno znati koje se supstancije koriste za gašenje pojedinih vrsta požara.

Ostale oblasti

PEŠAK U SAOBRAĆAJU

SAOBRAĆAJ 1 (spektrum 48K)
Slavka i Slobodan Ivanović
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Ovaj program namenjen je učenicima u I i II razredu osnovne škole i na zabavan i zanimljiv način uvodi najmlađe u problematiku saobraćajnog vaspitanja. Obrađene su sledeće metodske jedinice: prelazak pešaka preko ulice (sa pešačkim prelazom i bez njega, semafor, značenje pokreta i položaja saobraćajnog milicionera i kretanje pešaka po putu van naseljenog mesta. Po želji program može koristiti latinično ili ćirilično pismo. Osim toga, pored pisanih poruka, gotovo sva pitanja i puno komentara daju se i glasom.

BICIKLISTA U SAOBRAĆAJU

SAOBRAĆAJ 1 (spektrum 48K)
Slavka i Slobodan Ivanović
Zavod za udžbenike i nastavna sredstva i PGP RTB, Beograd (1986)

Program se sastoji iz dva dela: testa znanja neophodnog da bismo se biciklom uključili u saobraćaj i praktičnog dela — simulacije vožnje bicikla ulicom. Da bi se prešlo na „vožnju" potrebno je besprekorno rešiti test u kome se proverava znanje saobraćajnih znakova, rešavaju raskrsnice i proverava znanje saobraćajnih propisa.

EVOLUCIJA

EVOLUCIJA („erik nave 64)
ŠKD FORUM Ljubljana (1986)

Program Evolucija, na istoimenoj kaseti, jedan je od klasičnih računarskih programa tipa simulacije. Reč je o interesantnom problemu: kako će se razviti dve vrste čije postojanje zavisi od nekoliko spoljnih faktora ali i od međusobnog dejstva?

SUPER-ŠAH

SUPER-ŠAH (galaksija 6—8)
Milan Pavlićević i Ivan Gerenčer
Zavod za udžbenike i nastavna sredstva Beograd (1985)

Ovaj program namenjen je početnicima u drevnoj igri na 64 polja. Osim mogućnosti da pojedinac igra šah protiv „galaksije" nudi i mogućnost analiza pozicije, snimanja trenutne pozicije i učitavanje pozicije koje je ranije snimljena. Na raspolaganju su 5 nivoa igre od čega zavisi „jačina" računara i brzina igre.

ABECEDA

ABC, (spektrum 48K)
Istok Zupan
ŠKD FORUM, Ljubljana (1986)

Program je namenjen osnovnom opismenjavanju najmlađih. Na slikovit i zanimljiv način moguće je učenje slova i njihovo sastavljanje u reči (slovenački).

BESEDE

ABC (spektrum 48K)
Istok Zupan
ŠKD FORUM Ljubljana (1986)

Ovo je nastavak programa Abeceda. Pomoću njega se uči sastavljanje rečenice od pojedinačnih reči.

PERFECT BASE

(„komodor 64)

ZOTKS, Ljubljana (1986)

Program je namenjen radu sa bazom podataka.



Obrazovni centar „Iskra Delta" u Novoj Gorici

NIJE SVE U HARDVERU

U obrazovnom centru „Iskra Delte" u Novoj Gorici odavno je shvaćeno da se sa golim hardverom ne može ništa učiniti. Kupcu, uz mašine, treba dati i odgovarajući softver i treba ga — obučiti.

Pre dve godine „Iskra Delta" je u Novoj Gorici zapušteni hotel „Argonauti" u saradnji sa lokalnim hotelskim preduzećem i uz pomoć opštine i banaka pretvorila u veliki obrazovni centar.

U renovirani školski deo — hotel je pretrpeo manje prepravke — stižu korisnici Deltinih proizvoda iz cele zemlje. U strategiji ovog proizvođača računara obrazovanje ima izuzetan značaj. Po njima nije problem samo kako popuniti kapacitete već i kako računar opslužiti da iz njega „isisneš" sve što nudi. Zato se tamo kupci i spremaju za to.

Uz obuku pruža im se mogućnost da vide šta proizvodi „Delta", pa je značajna i tržišna funkcija obrazovnog centra u Novoj Gorici. Zato je u tom kraju „Iskra Delta" napravila dva jaka referentna centra. Jedan je u energetici: daljinsko upravljanje hidroelektranom na Soči (komandni centar je opremljen „Deltinim" mašinama), a drugi je zdravstveni informativni centar u Novoj Gorici.

Zahvaljujući velikoj frekvenciji ljudi kroz centar, uspeli su da krenu sa predavanjima iz rukovođenja. Ta druženja im omogućavaju i da prate potrebe širom zemlje, da bi kasnije okupili zainteresovane i predstavili im svoj program. Ko dolazi u školu „Iskra Delte" u Novoj Gorici?

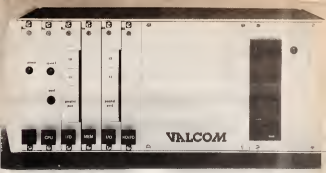
Pre svega kupci. Kada zaključi ugovor, u kome postoji član koji određuje koliko troškova za obrazovanje daje „Iskra Delta", kupac dobije bonove i katalog sa programima obrazovanja u centru. Sam bira termin, program koji će pohađati — naravno, u zavisnosti od opreme koju je kupio. Seminari traju različito. Po pravilu uči se od ponedeljka do petka. Ali, kursevi su tako podeljeni da se mogu nastavljati u neke veće „module" koji obrazuju operatere, programere, sistemske inženjere... Naravno, to je samo dopuna njihovih znanja.

Vrata centra otvorena su preko cele godine, sem mesec dana preko leta, kada je remont opreme. A nje ima zaista dosta. U računskom centru su mašine „Delte", u 22 učionice je 150 terminala (u svakoj učionici najviše 16 ljudi)... Iza toga stoje dva „Delte" računara 4860. Tu su i „partner", „triglav", a uskoro će biti i AT. Predavanja drže instruktori proizvođača, ali i drugi, jer je u Gorici samo 12 zaposlenih. Dolaze stručnjaci iz ljubljanskog dela „Delte", ali i ljudi iz „Digitala", jer nije retkost da stručnjaci „Iskra Delte" gostuju u „Digitalu".

Planovi rada centra postoje za šest meseci unapred, ali prva tri u tom planu su fiksirana, sigurna. Kada se odluče za termin, kupci se jave i time rezervišu mesto u hotelu obrazovnog centra i na samom kursu. Nastava traje od 9 ujutru do 13 časova, da bi se nastavila posle ručka. Grupu vodi instruktor, koji sa polaznicima živi, jede, prati ih i maltene ispunjava želje. „Ukoliko su polaznici povučeni", objašnjava Oton Mozetič, direktor osnovne jedinice „Iskra Delte", „u svakoj sobi se može postaviti terminal povezan sa našim računskim centrom. Ako su zadovoljni nastavom, u slobodnom vremenu se mogu opustiti u bazenu, na teniskim terenima, trim stazi... Najpopularnija rekreacija petkom je obilazak prodavnica u Italijanskoj Gorici."

Vrata centra su otvorena i za druge, ne samo za kupce. Školske centre, doduše manje, „Iskra Delta" ima i u većim gradovima u zemlji, koji su u nekoj hijerarhiji ispod ovog u Novoj Gorici. „Ako neki kupac ili u nekom delu zemlje ima većih potreba za specijalnim kursevima", kaže Mozetič, „onda šaljemo instruktore u bilo koji od ovih centara — Zagreb, Beograd, Novi Sad, Skoplje... Možemo ih imati i u fabrikama, tu smo dosta fleksibilni."

E. P.



VME—BUS NAPRAVITE SVOJ KOMPJUTER

Serija VALCOM VMEbua modula na veličini jednoatruke Evropa pločice!

GRAPH 1 — 2.988.000 din

JEDINICA GRAFIČKOG KONTROLERA VISOKE REZOLUCIJE

- 68348 ACRTC kontrolar sa modom dvostruko pristupa memoriji
- 800×600×4/50 Hz ili 720×540×4/60 Hz — »noninterface«
- 16 boja od 4096
- 3 razdvojana prikaza i jedan prozor
- 512K byte-a videomemorija — 1024×1024 akvivalantna rezolucija
- hardverski križni kursor i grafički kursor
- 1 do 16 hardverski zoom razdvojani vertikalni i horizontalni
- vertikalni i horizontalni »scroll«

RDMRAM — od 1,188.000 do 3,110.000 din

16-BITNA DINAMIČKA MEMORIJA SA PARITETOM

- do 2M byte-a dinamičkog RAM-a i do 256K byte-a EPROM-a
- 24 adresne linije
- 16 linija za podatke
- 350 ns vrijeme pristupa u čitanju i pisanju
- paritetno ispitivanja na nivou byte-a

MULTIFUNCTION CARD — od 936.000 do 1,051.000 din

VIŠEFUNKCIJSKA JEDINICA

- 2 serijska I/O kanala sa podržanim multiprotokolom (7201)
- izbor RS232/RS422 transciever-a
- ugradan sat (146818) aku-batarija na ploči
- 20 linijski paralalni port (6522) i 2 16-bitna timar-a
- 3 softverski programabilna kanala prakida (interrupt channal)

CPU68-1 — od 684.500 do 664.000 din

- 68000 CPU ili 68010 CPU (8MHz)
- 16 ili 64K byte-a statičkog RAM-a
- 2 EPROM podnožja za ROM-ove do 128K byte a
- funkcija sistemskog VME kontrolara
- kontrola 7 nivoa prekida

HD/FD/TAPE CDNTRDLLER — od 1,670.000 do 1,836.000 din

UNIVERZALNI KONTROLER ZA HARD DISK FLOPPY DISK I STREAMER

- kontrolira 4 jedinice za masovno pospremanja podataka
- kontrolira do 2 3½ i 5¼ Winchestar diska
- kontrolira do 4 3½ i 5¼" floppy diska
- kontrolira 1 streamar
- 16K byte-a privatna dual-port mamorije na ploči za brzi transfar

IEEE488/DMA&RS232 CARD — 1,080.000 din

IEEE-488 BUS KONTROLER SA SERIJSKIM I/O KANALOM

- IEEE-468 controllar/talkar/listanar mod sa čipom 9914
- brzi DMA transfar od 500 K byte/s
- asinhroni serijski I/O kanal sa programabilnom brzinom prijenosa
- 2 softvarski programabilna kanala prakida (interrupt channel)

MDTHERBDARD VAL/MDTH1 — 396.000 din

VMEbus FUNKCIONALNO EKVIVALENTNI MOTHERBOARD

- 9 utičnih mjesta jednostruka visina
- funkcionalno akvivalantan VMEbus P1 motharboardu
- 24 adresna linija i 16 linija za podatke
- svaka linija signala zaključena sa 330/470 otpornicima
- IACK i Bus-Grant daisy-chain signali između svakog utičnog mjesta
- standardna 19" dimenzija

Računalo je instalirano na fakultetima u Mariboru i Ljubljani.

SERVIS I IZRADA ELEKTRONIČKIH UREĐAJA

VALCOM

TRG SENJSKIH USKOKA 4
41020 ZAGREB
TEL 041/529-682 i 520-803

INSTITUT „MIHAJLO PUPIN“ — Beogradska

RAČUNARSTV



Tri decenije istraživanja i projektovanja u računarstvu Institut Mihajlo Pupin

Gotovo tri dacanije etručnjaci instituta „Mihajlo Pupin“ beva aa ietreživenjime, rezvojam i projaktovenjem eletema i uradaja iz obiaati računareka tahnika i informatika, u okviru Laboratorija za digitainu tehniku, kaenija nazvana OOUR Računaraka tahnike, odnosno RJ „RAČUNARSTVO“. U protekiih tridaaatak godina u iMP eu reaiizovani mnogi programi razvoja i proizvodnja hardvara i aoftvera iz četiri genaracije domaćih računara poznatih eerije CER, HRS-100 i TiM. Takođe ja bio značajen rad na razradi primana računarakih alatama za automataku obradu informacija i računarako upravljanja aloženim procesima i mašinama.

REFERENCE IMP

Najvažnija referance iz prošlog perioda su

(1) Digitalni računar sa elektronskim cevima CER-10 namenjen za statistička istraživanja za SIV (domaći računarski prvanac iz 1960. godine).
(2) Serija računara za poslovne obrada podataka CER-200 CER-22 i CER-12 koji su našli primenu u više jugoslovenskih preduzeća banaka i saveznih ustanova tokom 60-tih godina
(3) Specijalizovani digitalni računari posebne namene kao što su CER-11 CER-111 i DRS-160 — „KOSMOS“,
(4) Hibridni računski sistemi HRS-100 namenjeni za simulaciju i upravljanja složenim dinamičkim procesima (isporučeni Institutu za problama upravljanja AN SSSR Moskva Institutu za katalizu Novosibirsk i Institutu za termodinamiku Moskva. 1971—1983 godine),
(5) Serija hibridnih miniračunara IMP-3001 (za ETF Sarajevo MŠC Split Skoplje Suboticu i TVA Zagreb),
(6) Računari za rad u realnom vremenu u okviru namenskih sredstava za JNA, kao i tranažeri i simulatori za nastavne centre i obuku,
(7) Serija mikroračunarskih sistema TIM od šalterskog PTT računara TIM-100 do višeprocesorskog sistema TIM-600 (period 1984—1988),

Ostvarana je uspešna saradnja sa domaćom privredom vojnotehničkim ustanovama državnim organima naučnim i obrazovnim institucijama na implementaciji računarske tehnologije i primeni softverskih metoda i proizvoda. Realizacija serije domaćih računara dovela je do potrebe za razvojem sistemskog softvera za njihovu podršku dok je proučavanje primana računara i njihovo uvođenja u praksu snažno motivisalo razvoj aplikacionog softvera, naročito iz oblasti saobraćaja bankarstva elektroprivreda vodoprivrede i za potrebe JNA

PROGRAMSKI PRAVCI

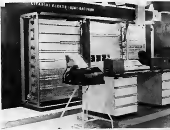
Današnja istraživanja iz računarske tehnike i informatike u IMP obuhvataju sledeće programske pravce

- razvoj novih arhitektura računara za paralelno procesiranje,
- višekorisnički i višeprocesorski računarski sistemi,
- računarska mreže (javne i lokalne),
- razvoj super mikroračunara na bazi µP od 32 bita,
- računarsko procesno upravljanje mašinama i sistemima,
- razvoj programskih sistema za rad u realnom vremenu,
- razvoj sistemskog i aplikacionog softvera,
- projektovanje informacionih sistema i distribuiranih baza podataka,
- primana veštačke inteligencija i razvoj ekspertnih sistema,
- modeliranja digitalno procesiranja signala i simulacije,
- projaktovanje i implamantacija računarskih sistema posebna namena
- razvoj tranažera, simulatora i drugih sistema za obuku u nastavnim centrima,
- razvoj elektronska oprama za računarako upravljanja saobraćajnom signalizacijom

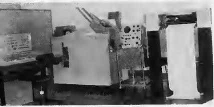
OD KONCEPCIJE DO REALIZACIJE SISTEMA

Institut MIHAILO PUPIN





Prvi domaći računar završen 1960 CER 10







Digitalni elektronski računar druge generacije CER-22 (Beogradska banka 1966)



Hibridni sistem treće generacije HRS-100 (Moskva 1971)





Originalne koncepcije računarskih sistema CER-12

Institut
MIHAILO
PUPIN

RESURSI INSTITUTA

Institut „Mihajlo Pupin" je, zahvaljujući brojnom naučno-istraživačkom kadru i savremeno opremljenim laboratorijama ostvario brz i dinamičan razvoj računarske tehnike. U proteklom periodu realizovan je veći broj računara iz serija CER, hibridnih računskih sistema HRS-100 i mikroračunarskih sistema TIM. Takođe su razvijani aplikacioni softver i složeni računarski programi za optimizaciju procesa u elektroenergetici, vodoprivredi, saobraćaju, vojnim primenama, industrijskoj robotici, mašinogradnji, telekomunikacionim mrežama i sistemima daljinskog nadzora, merenja i upravljanja.

U Institutu radi preko 300 istraživača sa VSS, a među njima više od 80 doktora nauka i magistara, sa ukupno 1200 zaposlenih. Približno jedna trećina istraživačkog kadra je usko vezana za stručne oblasti računarstva i informatike. Osim razrada idejnih rešenja i projektovanja softverskih i hardverskih računarskih proizvoda, stručnjaci Instituta pružaju i usluge u analizi i vrednovanju sistema, uvođenju računara i implementaciji IS, održavanju i servisiranju opreme, te obuci i sistemskoj podršci korisnika. Pored izrade brojnih laboratorijskih modela i prototipova, u Institutu su realizovana i domaće serija računara na bazi originalnih tehničkih rešenja i sopstvene konstruktorske dokumentacije.

Istraživački rad praćen je brojnim stručnim dokumentima, priručnicima i uputstvima o uvođenju i implamentaciji računarskih sistema u širokom spektru primena (matematički modeli, algoritmi, softverske rešenja, programski paketi, logička šema i slično). Biblioteka Instituta „Mihajlo Pupin" je jedna od najbogatijih u Jugoslaviji po tehničkim publikacijama, sa mnogim knjigama i časopisima iz oblasti računarstva i informacionih tehnologija. Na raspolaganju je veoma obimna interna dokumentacija (studije, stručne beleške, elaborati, idejni projekti, analize ostvarljivosti i pouzdanosti, ocene performansi sistema i slično).

Institutske laboratorije su bogato opremljene mernom instrumentacijom, razvojnom opremom i desetinama savremenih raču-



Fleksibilnost kod povezivanja u računarske mreže. TIM — 200G



Mikroračunarski sistemi zasnovani na VLSI tehnologiji. TIM — 011



Višekorisnički MS DOS sistemi TIM — 390

nara tipa TIM (sopstvena izrada) VAX Delta Honeywell Burroughs Wang Apple HP IBM-PC i drugi. Raznovrsnu perifernu opremu čine magnetni i optički diskovi diskete i traka štampači ploteri, konvertori grafički i video terminali razvojni sistemi i radne stanice, pripremne i pomoćne mašine i mnogi drugi uređaji

ČETIRI TEHNOLOŠKE GENERACIJE

(I) **CER-10** je računar koji pripada prvoj tehnološkoj generaciji zasnovanoj na elektronskim cevima. Namenjen je naučno-tehničkim kalkulacijama i statističkoj obradi podataka. U računar je ugrađeno 1 750 elektronskih cevi, 1 500 tranzistora, 120 000 magnetnih torusnih jezgara i 1 700 impulsnih transformatora. Potrošnja energije ove mašine bila je oko 10 kVA, a brzina rada iznosila je 50 000 operacija u sekundi. CER-10 je uspešno radio za potrebe SIV-a i Tanjuga, a kasnije u Elektrotehničkoj školi „Nikola Tesla" u Beogradu sve do 1970. godine. U to vreme Jugoslavija je sa sopstvenom koncepcijom razvoja računara bila u svropskom vrhu računarstva, jer tada je i u razvijenim zemljama Evrope postojalo svega nekoliko istraživačkih centara sa vlastitim računarskim konstrukcijama. Konstruktori „Cifarskog Elektronskog Računara" CER-10 su Rajko Tomović, Tihomir Aleksić, Ahmed Mandžić, Vukašin Masnikosa, Dušan Hristović, Petar Vrbavac i Milojko Marić.

(II) **Serija „CER" računara** zasnovana je na tehnologiji poluprovodnika (tranzistori i diode) i štampanih kola sa raznovrsnom perifernom opremom na bazi bušenih kartica, papirne trake i magnetnih diskova i traka. U periodu od 1962. do 1972. godine za čitavu seriju „CER" dominantni su bili kreativno stvaralaštvo domaćih projektanata i originalne koncepcije računarskih sistema. Najpoznatiji modeli bili su knjigovodstvena mašina CER-20, računari CER-200, CER-22, CER-11 i CER-12 (koji još i danas radi u ERC-u Instituta „Mihajlo Pupin")

(III) **Hibridni sistem HRS-100** zajedničko je delo stručnjaka Instituta „Mihajlo Pupin" iz Beograda i Instituta za probleme upravljanja (ANSSSR) iz Moskve. Završen je 1971. godine i spada u







najmodernije sistema III tehnološke generacija. HRS-100 je zaista objedinio svetska dostignuća iz domena analogne i digitalne računarske tehnologije, a sastoji se iz tri dela: digitalnog računara, linka i 4 analogna računara. U sistem je, pored ostalog, ugrađeno oko 3 500 integralnih kola raznog stepena složenosti povezanih tzv. wire-wrap tehnikom. Sa raznovrsnom perifernom opremom i analognim uređajima HRS-100 zauzima površinu od oko 80 m². Sistem programske podrške dat je na jezicima Fotran, ALMO, Autocod i Algol.

Hibridni računski sistem HRS-100 je predviđen za naučno-tehnička istraživanja u prirodnoj i ubrzanoj vremenskoj skali i za modeliranje složenih dinamičkih procesa. Njegova brzina računanja iznosi oko 500 000 operacija u sekundi. Uspešno je primenjivan za optimizaciju procesa i sistema u aerokosmonautici, elektroenergetici, naftnoj hemiji, tehnici elektroveza i biološko-medicinskim istraživanjima u više instituta AN SSSR u Moskvi i Novosibirsku.

(IV) **Mikroračunarski sistemi „TIM"** pripadaju najnovijoj generaciji računara, a zasnovani su na tzv. VLSI tehnologiji (mikroprocesori i druge mikroelektronske komponente izuzetno visokog stepena složenosti: jedan elektronski čip odgovara mnoštvu od nekoliko stotina hiljada ekvivalentnih tranzistora, a debljina međusobnih veza dostiže jedan mikrometar). Glavna odlika TIM računara su: domaći razvoj (vlastiti projekti i konstrukcija, maksimalni udeo domaćih komponenata), funkcionalna modularnost hardvera i softvera, standardizacija podsklopova i fleksibilnost kod povezivanja u računarske mreže. Zahvaljujući arhitekturi „otvorenog kraja" i širokom spektru modula i perifernih uređaja moguće je zadovoljiti zahteve raznovrsnih korisnika. Porodica TIM obuhvata sledeće modele: TIM 011 i TIM 020 — školski računari, TIM 100 — šalterski računari za pošte, banke i slične ustanove, TIM 200 — terminalski orijentisani računari opšte namene, TIM 200G — računari za grafičku industriju (priprema tekstova za fotoslog), TIM 300 — računari za procesno upravljanje, TIM 350 — višeprocesorski bankarski sistemi, TIM 390 — višekorisnički MS DOS sistemi, TIM 400 — višekorisnički Xenix sistemi i TIM 600 — super mikroračunari sa 32-bitskom arhitekturom.

Dušan Hristović, dipl. inž.

SVE „TIMOVE" NAREDBE

Naredbe i komande

AUTO — **AUTO [<broj linije> [,<uvećanje>]]**

Automatsko zadavanje broja programske linije pri unošenju programa

CALL — **CALL <ime potprograma> [(<lista promenljivih>)]**

Pozivanje mašinskog potprograma.

CHAIN — **CHAIN [MERGE]<ime programa> [,[<izraz za br.lin.>] [,ALL] [,DELETE <opseg>]]**

Pozivanje (mešanje) više programa sa čuvanjem ili bez čuvanja promenljivih

CLEAR — **CLEAR [,[<izraz1>] [,<izraz2>]]**

Postavlja sve numeričke promenljive na nulu sve znakove promenljive na prazno zatvara sve otvorene datoteke i opciono postavlja kraj raspoloživa memorije i dubinu bejzik steke

CLOSE — **CLOSE [[#]<broj datoteke> [,[#]<broj datoteke>]]**

Zatvara zadate ili sve otvorene datoteke

CLS — **CLS**

Brisanje ekrana (grafičkog i tekstualnog)

COMMON — **COMMON <lista promenljivih>**

Čuvanja vrednosti promenljivih iz <liste promenljivih> pri pozivanju novog programa (vidi instrukciju CHAIN)

CONT — **CONT**

Nastavak izvršavanja programa posle prekida.

DATA — **DATA <lista konstanti>**

Čuvanje podataka u programu

DEF FN — **DEF FN <ime> [(<lista promenljivih>)] = <definicija funkcije>**

Definisanje nove funkcije

DEFINT DEFSNG DEFDBL DEFSTR — **DEF<tip> <promenljive>**
DEF<tip> <opseg promenljivih>
<tip> = (INT, SNG, DBL ili STR)

Definisanje tipa promenljive.

DEF USR — **DEF USR<broj 0 .9>=<celobrojni izraz>**

Definisanje adrese početka mašinskog potprograma (funkcije).

DELETE — **DELETE [<broj linije1>] [−<broj linije2>]**

Brisanje programske linije ili grupe linije

DIM — **DIM <lista nizova>**

Rezervisanje mesta u memoriji za nizove postavljanje maksimalnog člana niza i dodeljivanje nulte vrednosti svim članovima niza

DRAW — **DRAW <x koord.>,<y koord.>[,<aps/rel>[,<intenzitet>]]**

Crtanje linije od pozicija grafičkog kursora do tačke zadate apsolutnim koordinatama crtanje vektora zadatog priraštajima u X i Y pravcu računatih od grafičkog kursora zadavanje intenziteta linije

EDIT — **EDIT <broj linije>**

Izmena sadržaja programske linije

ELIPSE — **ELIPSE <radijus x> [,<radijus y> [,<intenzitet>]]**

Crtanje elipse ili kruga zadatog intenziteta sa centrom na poziciji grafičkog kursora.

END — **END**

Prekid izvršavanje programa, zatvaranje svih datoteka i povratak u komandni način rada.

ERASE — **ERASE <lista nizova>**

Oslobađanje memorijskog prostora zauzetog nizovima čija su imena navedena u <list nizova>

ERR i ERL promenljive

Promenljiva u izrazu
informacije o nastaloj grešci

ERROR — **ERROR <celobrojni izraz>**

1) Simuliranje pojavljivanja greške
2) Definisanje kodova za nove greške (koje se mogu javiti pri upotrebi programa)

FIELD — **FIELD [#]<broj datoteke>,<dužina polja> AS <znakovna promenljiva>...**

Definisanje mesta u memoriji (u daljem tekstu bafer) za formiranje jednog zapisa datoteke sa slučajnim pristupom

FILES — **FILES [<ime datoteke>]**

Prikazuje imena datoteka na (zadatom) disku

FILL — **FILL <x koord.>,<y koord.> [,<aps/rel>[,<intenzitet>]]**

Popunjavanje površine grafičkog ekrana do granice omeđene zadatim intenzitetom

FOR_NEXT — **FOR <np>=<i1> TO <i2> [STEP <i3>]**

—
— (telo petlje)
NEXT [<np>] [,<np>]
<np>=numerička promenljiva
<i1> <i2> <i3>=numerički izrazi
Omogućiti da se izvestan broj instrukcija (telo petlje) ponavlja tačno određen broj puta.

GET — **GET [#]<br. datoteke>[,<broj zapisa>]**

Čitanje zapisa u bafer iz datoteke sa slučajnim pristupom

GOSUB — RETURN — **GOSUB <br linije>**

<br linija>početak potprograma
—
RETURN
Skok u potprogram (GOSUB) i povratak iz potprograma (RETURN)

GOTO — **GOTO <br. linije>**

Bezuslovni skok na zadatu programsku liniju.

IF .. THEN — ELSE i IF — GOTO

1 IF <iz.> THEN <ins>/<br. lin.>[ELSE <ins>/<br lin.>]
2 IF <iz.> GOTO <br lin.> [ELSE <ins>/<br lin.>]

<A> / <B> ⇒ ili A ili B (upotrebljava se jedno od ponuđenih)
<iz> = aritmetički ili logički izraz
<ins> = instrukcija ili instrukcije
<br lin> = broj programske linije
Na osnovu izraza <iz> odlučuje se o daljem razvoju programskog toka (uslovni skok)

INPUT — **INPUT [;] [<poruka>,] <lista promenljivih>**

Omogućiti unos podataka sa tastature za vreme rada programa

INPUT# **INPUT# <br. datoteke>, <lista promenljivih>**
Čitanje podataka iz sekvencijalne datoteke ili bafera za datoteku sa slučajnim pristupom i dodeljivanje podataka promenljivima u <listi promenljivih>

KILL **KILL <ime datoteke>**
Brisanje datoteke (programa) sa diska

LINE INPUT **LINE INPUT [;] [<poruka>,] <znakovna promenljiva>**
Unos velikih grupa znakova (do 254 znaka) u znakovnu promenljivu

LINE INPUT# **LINE INPUT#<br. datoteke>,<znakovna promenljiva>**
Unos velikih grupa znakova (do 254 znaka) iz sekvencijalne datoteke (sa diska) u znakovnu promenljivu

LIST **LIST [<broj linije> [−[<broj linije>]]]**
Prikazivanje na ekranu programa ili dela programa koji je u memoriji

LLIST **LLIST [<br linije>[−[<br linije>]]]**
Štampanje programa ili dela programa, koji je u memoriji na štampaču

LOAD **LOAD <program>[,R]**
Učitavanje programa sa diska u programsku memoriju

LPRINT i LPRINT USING
LPRINT [<lista izraza>]
LPRINT USING <znakovni izraz>,<lista izraza>
Štampanje podataka iz programa na štampaču sa formatiranjem i bez njega

LSET i RSET
LSET <znakovna promenljiva>=<znakovni izraz>
RSET <znakovna promenljiva>=<znakovni izraz>
Upisivanje poravnatih podataka u bafer za datoteku sa slučajnim pristupom
Priprema podataka za snimanje na disk (vidi instrukciju PUT)

MERGE **MERGE <ime programa>**
Omogućava mešanje programa sa diska i programa u programskoj memoriji

MID$** **MID$(<zi1>,<p>[,<k>])=<zi2>
<zi1> <zi2>=znakovni izrazi
<p> <k>=celobrojni izrazi
Zamena dela grupe znakova drugom grupom znakova u znakovnim izrazima.

MOVE **MOVE <x koord.>,<y koord.>[,<aps/rel>[,<intenzitet>]]**
Postavljanje nove pozicije grafičkog kursora na zadate koordinate.
Postavljanje boje

NAME **NAME <staro> AS <novo>**
Promena imena datoteke na disku

NEW **NEW**
Brisanje programa i promenljivih iz memorije

ON ERROR GOTO **ON EFFOR GOTO <br linije>**
Omogućuje da se pri pojavi greške umesto prekida programa izvrši potprogram koji obrađuje grešku

ON .. GOSUB i ON ... GOTO
ON <izraz> GOTO <br linije>, <br linije>, <br linije> ...
ON <izraz> GOSUB <br linije>, <br linije>, <br linije> ...
Skok na <br linije> u zavisnosti od rezultata <izraza>. Višestruko grananje ili razgranati poziv potprograma.

OPEN **OPEN <način>,[#]<broj datoteke>,<ime datoteke>[,<duž.zapisa>]**
Otvaranje datoteke za rad

OUT **OUT <i>,<j>**
gde su <i> i <j> celobrojni izrazi
Slanje jednog bajta na U/I (ulazno/izlazni) port mikroprocesora.

PAINT **PAINT <x koord.>,<y koord.>[,<aps/rel>[,<intenzitet>]]**
Promena boje površine

PLOT **PLOT <x koord.>,<y koord.> [,<aps/rel> [,<intenzitet>]]**
Crtanje tačke i postavljanje pozicije grafičkog kursora.

POKE **POKE <i>,<j>**
gde su <i> i <j> celobrojni izrazi
Upis bajta na zadatu memorijsku lokaciju

PRINT **PRINT [<lista izraza>]**
Ispisivanje podataka na ekran

PRINT USING **PRINT USING <format>;<lista izraza>**
Ispisivanje numeričkih i znakovnih vrednosti koristeći *(using)* zadati format ispisa.

PRINT# i PRINT# USING
PRINT#<broj datoteke>, [USING<format>,]<lista izraza>
Ispis vrednosti (podataka) u sekvencijalnu datoteku

PUT **PUT [#]<broj datoteke>[,<broj zapisa>]**
Upis zapisa iz bafera u datoteku sa slučajnim pristupom

RANDOMIZE **RANDOMIZE [<celobrojni izraz>]**
Inicijalizacija generatora slučajnih brojeva

READ **READ <lista promenljivih>**
Čitanje podataka iz DATA liste i dodeljivanje tih vrednosti promenljivima

REM **REM <tekst primedbe>**
Komentarisanje programa ili odvajanje funkcionalnih celina programa objašnjenjima.

RENUM **RENUM [[<novi broj>][,[<stari broj>][,<uvećanje>]]]**
Renumeracija programskih linija

RESET **RESET**
Zatvaranje svih otvorenih datoteka i snimanje direktorijuma na disk pre no što se izvadi disketa

RESTORE **RESTORE [<br linije>]**
Inicijalizacija DATA liste za čitanje

RESUME
RESUME
RESUME 0
RESUME NEXT
RESUME <br linije>
Nastavak rada programa po obradi greške

RUN
RUN [<broj linije>]
RUN <ime programa>[,R]
Izvršavanje programa u memoriji uz eventualno prethodno učitavanje sa diska

SAVE **SAVE <ime programa>[,A/,P]**
Upisivanje programa iz memorije u datoteku

SOUND **SOUND <visina>,<trajanje>**
Generisanje zvuka

STOP **STOP**
Prekid izvršavanja programa i povratak u komandni način rada

SWAP **SWAP <promenljiva>, <promenljiva>**
Međusobna zamena vrednosti dve promenljive.

SYSTEM **SYSTEM**
Kraj rada u bejziku i povratak u operativni sistem

TEXT **TEXT <zi>,<x koord.>,<y koord.>[,<aps/rel>[,<intenzitet>]]**
gde je <zi> znakovni izraz
Ispisivanje teksta na proizvoljnoj koordinati na ekranu

TRON/TROFF
TRON
TROFF
Uključuje/isključuje ispis brojeva programskih linija koje se izvršavaju

WHILE ... WEND **WHILE <izraz>**
—
— [<telo petlje>]
—
WEND
Izvršavanje serije instrukcija dok je <izraz> tačan (tj. različit od nule).

WIDTH **WIDTH [LPRINT] <celobrojni izraz>**
Postavljanje širine ispisa na ekranu ili štampaču

WRITE **WRITE [<lista izraza>]**
Ispis podataka na ekran

WRITE# **WRITE#<broj datoteke>,<lista izraza>**
Ispisivanje podataka u sekvencijalnu datoteku

Funkcije

ABS **ABS(X)**
Vraća apsolutnu vrednost izraza X

ASC **ASC(X$)**

Vraća numeričku vrednost aski kod prvog znaka iz rezultata znakovnog izraza X$ Ako je rezultat izraza X$ NULL znak (znak sa aski kodom 0) ispisuje se Illegal function call (Nedozvoljen poziv funkcije)

ATN **ATN(X)**

Vraća vrednost funkcije ARKUS TANGENS izraza X zadatog u radijanima Rezultat funkcije ATN je uvek u opsegu — PI/2 do PI/2 Rezultat izraza X može biti ma koji numerički tip a rezultat funkcije ATN je uvek u jednostrukoj preciznosti

CDSL **CDSL(X)**

Pretvara vrednost izraza X u vrednost dvostruke preciznosti

CNR$** **CHH$(I)

Vraća znak čiji je aski kod vrednost izraza I CHR$ se često upotrebljava za slanje nekog znaka koji se ne prikazuje na ekranu (štampaču) ali vrši neku funkciju

CINT **CINT(X)**

Vraća vrednost X zaokruženu na bliži ceo broj Ako X nije u opsegu −32768 do 32767 ispisuje se greška Overflow (prekoračenje)

COS **COS(X)**

Vraća kosinus ugla X zadatog u radijanima Rezultat funkcije COS je u jednostrukoj preciznosti

CSNG **CSNG(X)**

Pretvara vrednost X u vrednost jednostruke preciznosti

CVI, CVS, CVD
CVI (<grupa znakova dužine 2 bajta>)
CVS (<grupa znakova dužine 4 bajta>)
CVD (<grupa znakova dužine 8 bajtova>)

Pretvaranje grupe znakova pročitanih iz datoteke sa slučajnim pristupom i numeričke vrednosti CVI pretvara 2 bajta u celobrojnu vrednost CVS 4 bajta u vrednost jednostruke i CVD 8 bajtova u vrednost dvostruke preciznosti

EOF **$DF(<broj datoteke>)**

Vraća −1 (tačno) kada je dostignut kraj datoteke Upotrebite EOF za testiranje kraja pri čitanju datoteke da izbegnete grešku Input past end (Čitanje posle kraja) EOF radi i sa datotekama se slučajnim pristupom GET posle kraja datoteke se slučajnim pristupom prouzrokuje da EOF bude −1 EOF se može upotrebljavati za nalaženje dužine datoteke se slučajnim pristupom ili za binarno pretraživanje istih

EXP **EXP(X)**

Vraća vrednost stepena osnove prirodnog logaritma (osnovna je na stepen X) X mora biti manje ili jednako 87 3365 U suprotnom ispisuje se Overflow greška (prekoračenje) rezultat je mašinska beskonačnost a izvršavanje programa se nastavlja

FIX **FIX(X)**

Vraća celobrojni deo bliži nuli Funkcija FIX je ekvivalentna izrazu SGN(X)*INT(ABS(X))

FRE **FH8 (0)**

FRE (" ")

NEX$** **NEX$(X)

Vraća znakovnu vrednost koja je heksadecimalni zapis vrednosti X Ukoliko X nije celobrojna vrednost vrši se zaokruživanje

INKEY$** **INKEY$

Vraća znak koji je pritisnut na tastaturi Ukoliko ništa nije otkucano vraća znak NULL (znak se aski kodom nula) Primljeni znak se ne prikazuje na ekranu Svi znaci se tastature mogu biti rezultat funkcija osim CTRL C koji prekida rad programa

INP **INP (I)**

Vraća celobrojnu vrednost (bajt) pročitanu se U/I porta I može imati vrednost iz opsega [0,65535]

INPUT$** **INPUT$(X[,[#]Y])

Vraća grupu znakova dužine X pročitanu sa tastature ili iz datoteke pod brojem Y Ako je za ulaz iskorišćena tastatura znaci koji se kucaju ne prikazuju se na ekranu Svi znaci mogu biti rezultat funkcije INPUT$ osim CTRL C koji prekida rad programa.

INSTR **INSTR ([I,]X$,Y$)**

Pretražuje Y$ da nađe pojavljivanje X$ u Y$ i vraća poziciju X$ u Y$. Opsano I je pozicija od koje će se pretraživati (ako nije navedeno I je jednakao 1) I mora biti u opsegu 1 do 255 a ukoliko je 0 ispisuje se greška Illegal argument in line XXXXX (Nedozvoljena vrednost argumenta u liniji XXXXX)
Ako je I>LEN(X$) X$ prazan ili Y$ ne postoji INSTR vraća vrednost 0.
Ako je Y$ prazan rezultat INSTR je I ili 1

INT **INT(X)**

Zaokružuje X na manji ceo broj

LEFT$** **LEFTS (X$,I)

Vraća prvih I znakova iz X$ Ako je I LEN(X$) ceo X$ je rezultat

LEN **L9N(X$)**

Vraća broj znakova sadržanih u X$. Broje se svi znaci uključujući i one koji se ne mogu ispisati

LOC **LOC (<broj datoteke>)**

Za datoteke se slučajnim pristupom
LOC vraća broj zapisa koji je upravo pročitan ili upisan (pomoću instrukcija GET ili PUT). Ako je datoteka otvorena a nije joj pristupano radi čitanja ili pisanja LOC vraća 0
Za sekvencijalne datoteke
LOC vraća broj sektora (1 sektor=128 bajtova) pročitanih ili upisanih od otvaranja datoteke (zavisno da li je ulazna ili izlazna datoteke vidi OPEN)

LOF **LDP(<broj datoteke>)**

Vraća broj sektora zapisenog u poslednjoj ekstenziji opisa smeštanja datoteke Ovaj opis se nalazi u direktorijumu Ako datoteka nije veća od 1 ekstenzije (1 ekstenzija=128 sektora=16 Kb) tada LOF vraća pravu dužinu datoteke (u sektorima)

LOG **LDG(X)**

Vraća logaritam X za osnovu e (prirodni logaritam) X mora biti veće od 0.

LPOS **LPDS(X)**

Vraća poziciju glave na štampaču X nema nikakvu funkciju

MID$** **MIDS(X$,I[,J])

Vraća J znakova iz X$ počevši od I-te pozicije I i J moraju biti u opsegu od 1 do 255 Ako je I=0 ispisuje se Illegal argument in line XXXXX (Nedozvoljena vrednost u liniji XXXXX) Ukoliko od I-te pozicije ima manje od J znakova J se ignoriše i uzimaju se svi znakovi od I-te pozicije do kraja grupe znakova Ako je I>LEN(X$) MID$ vraća praznu grupu znakova (dužine 0)

MKI$, MKS$, MKD$
MKI$(<celobrojni izraz>)
MKS$(<izraz jednostruke preciznosti>)
MKD$(<izraz dvostruke preciznosti>)

Pretvara numeričke vrednosti u grupe znakova Numeričke vrednosti pre stavljanje u bafer za datoteku sa slučajnim pristupom moraju biti pretvorene u znakove (zbog LSET i RSET instrukcija) MKI$ pretvara celobrojnu vrednost u grupu od 2 znaka (2 bajta) MKS$ jednostruku preciznost u 4 znaka a MKD dvostruku u 8 znakova.

OCT$ **OCTS**

Vraća grupu znakova koja predstavlja vrednost X u oktalnom brojnom sistemu Ukoliko X nije ceo broj biće zaokružen

PEEK **PE$K(I)**

Vraća vrednost I te memorijske lokacije Rezultat je u opsegu od 0 do 255 a I može imati vrednost od 0 do 65535

POS **PDS(I)**

Vraća poziciju kursora u okviru linije Krajnja leva pozicija je 1 I ignoriše se

RIGNT$** **HIGHS(X$,I)

Vraća poslednjih I znakova iz X$ Ako je I>=LEN(X$) vraća X$ a ako je I=0 vraća se prazna grupa znakova (dužine 0)

RND **HND[(X)]**

Vraća slučajni broj između 0 i 1 Sekvencija slučajnih brojeva se ponavlja istim redosledom ako nije upotrebljena instrukcija RANDOMIZE Ako je X<0 ponavlje se sekvencija od početka za X=0 ponavlja se prethodni slučajni broj a ako je X>0 ili izostavljeno rezultat je sledeći slučajni broj sekvence

SGN **9GN(X)**

ako je X>0 rezultat je 1
ako je X=0 rezultat je 0.
ako je X>0 rezultat je −1

SIN(X)

Vraća sinus ugla X zadatog u radijanima Rezultat funkcije SIN je jednostruka preciznost

SPACE$** **SPAC$$(X)

Vraća X blanko znakova. Ako X nije ceo broj biće zaokružen X može imati vrednost od 0 do 255

SPC **SPC(I)**

Štampa I blanko znakova SPC može biti upotrebljen samo uz instrukcije

PRINT i LPRINT. I može biti od 0 do 255. UZ SPC se podrazumeva znak tačka-zarez (;) na kraju

SQR — **SQR(X)**

Vraća kvadratni koren vrednosti X. X mora biti>0.

STR$** — **STR$(X)

Vraća znakovnu reprezentaciju vrednosti X (pretvara brojeve u grupu znakova)

STRING$** — **STRING$(I,J) / **STRING$(I,X$)**

Prvi format vraća I znakova sa aski kodom J.
Drugi format vraća prvi znak iz X$.

TAB — **TAB(I)**

Pozicionira kursor u I-tu kolonu. Ako je I manje od trenutnog položaja kursor se postavlja u sledeću liniju na I-tu poziciju. I može imati vrednost od 1 do 255. TAB se upotrebljava samo u instrukcijama PRINT i LPRINT.

TAN — **TAN(X)**

Vraća tangens ugla X zadatog u radijanima. Rezultat funkcije TAN je jednostruke preciznosti. Ako dođe do prekoračenja ispisuje se poruka Overflow (prekoračenje), rezultat je mašinska beskonačnost sa odgovarajućim predznakom, a izvršavanje programa se nastavlja.

USR — **USR[<broj>] (X)**

Poziva mašinski potprogram i prosleđuje mu vrednost X. <broj> je cifra između 0 i 9. Podrazumeva se 0 ako drugačije nije navedeno. Adresa mašinskog potprograma definisana je u DEF USR instrukciji.

VAL — **VAL(X$)**

Vraća numeričku vrednost grupe znakova koji predstavljaju numeričku konstantu. VAL ignoriše blanko, CR, LF i TAB znakove.

VARPTR — **1) VARPTR(<ime promenljive>)** / **2) VARPTR(#<broj datoteke>)**

1) Vraća adresu prvog bajta podatka promenljive. Promenljiva mora postojati tj. mora joj se pre poziva ove funkcije dodeliti neka vrednost, u suprotnom ispisuje se greška Illegal function call (Nedozvoljen poziv funkcije). VARPTR vraća vrednost u opsegu −32768 do 32767. Za znakovne promenljive VARPTR vraća adresu na kojoj se nalazi dužina sadržaja promenljive. Adresa prvog znaka sadržaja promenljive nalazi se na memorijskim lokacijama koje slede.

Kontrolni znaci

Bejzik prihvata sve znake engleske abecede (A-Z,a-z), specijalne znake (znake iznad brojeva i na desnoj strani tastature) i određene kontrolne znake. Kontrolni znaci se dobijaju pritiskom na određene tastere ili kombinacijom CTRL (kontrol tastera) i nekog slova. Bejzik prepoznaje sledeće kontrolne znake.

ZNAK		TASTER	OBJAŠNJENJE / FUNKCIJA
CTRL	A		Menjanje teksta komandne linije
CRTL	C	BRK	Prekida red programa i vraća se u komandnu liniju
CTRL	H	BS	Pomera kursor jedno mesto ulevo i briše znak pod njim
CTRL	I	TAB	Tabulator (pomera kursor za 8 znakovnih mesta udesno)
CTRL	J	LF	Pomera kursor u sledeću fizičku liniju ekrana
CTRL	M	RET	Kraj komandi, kraj programskog reda, novi red. RET se u aski kodu označava sa CR
CTRL	O		Isključuje štampanje dok program radi (naredno pritiskanje uključuje štampanje)
CTRL	R		Ponovo ispisuje liniju koja se kuca
CTRL	S		Zaustavlja program
CTRL	Q		Nastavlja izvršavanje programa zaustavljenog sa CTRL S
CTRL	U		Briše liniju koja se kuca
CTRL	[	ESC	Izlaz iz izmena teksta programske linija
DEL		DEL	Briše poslednji otkucani znak i stavlja ga između kosih crta

mladinska knjiga
knjigarne in papirnice

NOVO U KNJIŽARAMA MLADINSKE KNJIGE
priručnici, udžbenici, programi...



PRIRUČNICI ZA RAČUNARE

Atari

ATARI 800 XL priručnik za rukovanje (sh)	8 500 din
ATARI 1040 ST priručnik za rukovanje (sh)	7 000 din
STEVE priročnik (slov)	13 000 din
ABC za Atari	18 000 din

Amstrad-Schneider

INTRODUCING AMSTRAD CPC 464 MACHINE CODE (engl)	4 000 din
PRACTICAL PROGRAMS FOR THE CPC 464 (engl)	4 000 din
Zarić AMSTRAD-SCHNEIDER CPC 464 priručnik (sh)	4 000 din
AMSTRAD CPC 464-PROGRAMIRANJE U ASEMBLERU (sh)	5 000 din
AMSTRAD CPC 464 664 6128 — PRIMENE (sh)	5 100 din
AMSTRAD CPC 6128 — priručnik (sh)	5 000 din

Commodore

OSNOVE PROGRAMIRANJA C64 (slov)	6 000 din
COMMODORE 64 — PROGRAMIRANJE NA LAK NAČIN (sh)	13 000 din
BASIC ZA MIKRORAČUNARE C 64 (sh)	3 700 din
ŠTA MOŽE COMMODORE 64 (sh)	7 350 din
MAŠINSKE RUTINE ZA VAŠ C 64 (sh)	9 700 din
Šolajić COMMODORE 64 — MEMORIJSKE LOKACIJE (sh)	4 500 din
COMMODORE 64 ROM S REVEALED (engl)	4 500 din
ADVANCED MACHINE CODE FOR THE C 64 (engl)	2 200 din
C 64 — DISK SYSTEMS AND PRINTERS (engl)	1 500 din
C 64 — USEFUL SUBROUTINES AND UTILITIES (engl)	1 8 0 din
COMMODORE 128 priručnik (sh)	13 000 din
Šolajić Zarić COMMODORE 128 priručnik za rad (sh)	5 000 din
COMMODORE ZA SVA VREMENA (sh)	18 000 din
C 64 128 — kurs asemblerskog programiranja (sh)	5 000 din

IBM PC

ABC PC AT/XT u 25 lekcija (sh)	9 400 din
Životić ABC PC (sh)	6 000 din
Špiler OSEBNI RAČUNALNIK PC/XT/AT (slov)	14 000 din

Navedene knjige i kasete možete da kupite odnosno poručite u knjižarama i prodavnicama papira Mladinske knjige. Narudžbine pouzećem. Priloženu narudžbenicu popunite i pošaljite na adresu

MLADINSKA KNJIGA — KIP, grosistička prodaja knjiga, 61000 Ljubljana, Titova 3 tel (061) 211-860

Oric

ORIC AND ATMOS MACHINE CODE (engl)	4 500 din
THE ATMOS PROGRAMMER (engl)	4 500 din
THE ATMOS BOOK OF GAMES (engl)	4 500 din
40 EDUCATIONAL GAMES FOR THE ORIC ATMOS (engl)	4 500 din

ZX spectrum

SPEKTRUM PRIRUČNIK (sh)	4 200 din
ZX SPECTRUM — PROGRAMIRANJE U BASIC-u (sh)	1 750 din
THE COMPLETE SPECTRUM (engl)	3 900 din
SPECTRUM GAMESMASTER (engl)	1 600 din
THE SPECTRUM BOOK OF GAMES (engl)	1 500 din
THE ZX SPECTRUM AND HOW TO GET THE MOST OF IT (engl)	1 500 din
SPECTRUM GRAPHICS AND SOUND (engl)	1 750 din
AN EXPERT GUIDE TO THE SPECTRUM (engl)	1 800 din

PROGRAMSKI JEZICI, PROGRAMIRANJE

STROJNI JEZIK ZA PROCESOR Z80 (slov)	5 000 din
MAŠINSKO PROGRAMIRANJE NA MIKROPROCESORIMA Z80 i 6502 (sh)	6 000 din
LOGO PROGRAMSKI JEZIK (sh)	2 250 din
INTRODUCING LOGO (engl)	2 900 din
Špiler BASIC (sh)	4 000 din
Dovedan BASIC — JEZIK I PROGRAMIRANJE (sh)	6 000 din
ZBIRKA ZADATAKA U BASICU (sh)	6 000 din
BASIC U NASTAVI MATEMATIKE (sh)	6 000 din
PASCAL priručnik (sh)	19 000 din
PASCAL — zbirka rešenih zadataka (sh)	10 250 din
TURBO PASCAL 3 0 (sh)	6 000 din
APLIKACIONI PROGRAMI IBM PC APPLE IIc (sh)	9 700 din
PC WORDSTAR — obrada teksta (sh)	11 800 din
KOMPJUTERSKA GRAFIKA (sh)	16 000 din
RAČUNARI I KOMUNIKACIJE (sh)	14 500 din
Turk PROGRAMSKI JEZIK C (slov)	5 000 din
COBOL programiranje u praksi (sh)	3 650 din
UNIX — KAKO GA KORISTITI (sh)	5 000 din
PROGRAMIRANJE ZA POČETNIKE	10 000 din
KUĆNI KOMPJUTERI—ALGORITMI I PROGRAMI (sh)	2 700 din
NUMERIČKI METODI ZA MIKRORAČUNARE (sh)	2 300 din
VIDEO KOMPJUTERSKE IGRE (sh)	2 300 din
ODRŽAVANJE I OPRAVKA KUĆNIH RAČUNARA (sh)	3 000 din
Kodek MIKROPROCESORJI delovanje in oporaba (slov)	8 000 din
RAČUNALNIŠKI SLOVAR (slov)	6 500 din
RAČUNARSKI REČNIK (sh)	1 200 din
FORTRAN 77 (slov)	18 000 din

KASETE S PROGRAMIMA ZA ZX SPECTRUM

MAČEK MURI ŠTEJE IN RAČUNA (slov in sh)	900 din
DOBER DAN MATEMATIKA (slov)	1 300 din
LOGIKA ZA STARŠE (slov)	1 300 din

NAPOMENA Cene navedene uz pojedine knjige bile su na snazi u početku novembra meseca. O cenama knjiga ne odlučuje prodavac Mladinska knjiga, nego ih podižu izdavači. Zato se izvinjavamo za eventualne nesporazume. Naručene knjige isporučujemo po cenama koje budu na snazi na dan narudžbe

NARUDŽBENICA — R.V.Š.

Potpisani (ime i prezime) ____________________

Tačna adresa (ulica mesto broj pošte) ____________________

____

Neopozivo poručujem — pouzećem — plaćću kod preuzimanja pošiljke — sledeće knjige kasete ____________________

____

____

Datum   Potpis

mladinska knjiga
knjigarne in papirnice

IZDAVAČKA RADNA ORGANIZACIJA
„GRAĐEVINSKA KNJIGA"

11000 BEOGRAD
Trg Marksa i Engelsa 8/II

40 GODINA



1. PRIRUČNIK ZA TIM — 011 BASIC, 118 strana, format 20x28 cm, povez broširan, cena 7 000 dinara
2. PRIRUČNIK ZA TIM — 011 DOS, oko 100 strana, format 20x28 cm, povez broširan — u štampi

U toku je isporuka školskih računara TIM — 011 instituta Mihajlo Pupin srednjim školama. Uz računar je pripremljen **Priručnik sa TIM — 011 BASIC,** u kome su opisane mogućnosti BASIC interpretatora proširanog grafičkim instrukcijama. Priručnik je podeljen u pet poglavlja: Uvod, Opšte karakteristike, Sastav BASIC-a, BASIC komande i instrukcije i BASIC funkcije. U dodacima su opisani rad sa datotekama, izvedene funkcije, poruke o greškama, a tu je i obavezna ASCII tabela.

U štampi je **Priručnik za TIM — 011 DOS,** u kome je predstavljen ZDOS, operativni sistem računara TIM — 011, koji je značajna nadgradnja CP4M-a.

3. McGraw — Hill
TERMINOLOŠKI REČNIK, RAČUNARI, ELEKTRONIKA
Autor: Sybil Parker
Pretplatna cena: 49 600 dinara, važi do 30.04.1988.

Na oko 500 strana — englesko-srpskohrvatski rečnik termina sa širim objašnjenjem iz oblasti računara i elektronike. Rečnik sadrži srpskohrvatsko-engleski registar svih termina.

Rečnik je namenjen korisnicima računara i računarskih sistema u bankama, organima uprave, vojsci, pošti, institutima, te programerima, projektantima, inženjerima, studentima, operaterima, nastavnicima, dacima i svim početnicima koji se prvi put upuštaju u ovu široku oblast.

4. **METOD KONAČNIH ELEMENATA U BASIC-U**
Autor: dr Milisav Kalajdžić

cena 22 000

Uz knjigu se može naručiti
a) kaseta sa programima — 7 000 dinara
b) disketa sa programima — 15 000 dinara
c) grafik disketa — 15 000 dinara

Ova knjiga predstavlja prvi potpun profesionalni software za proračun mašinskih i drugih konstrukcija na mikroračunarima.

Knjiga sadrži opis 16 MEKELBA konačnih elemenata i definisanja sistema modela konstrukcije, 18 programa napisanih u BASIC-u koji su testirani (priloženo 34 test-primera). Svi programi su odštampani na preko 6 300 redova.

Svi MEKELBA programi snimljeni su na kasetu ili disketu. Dodatni MEKELBA GRAFIK software sadrži tzv. HIRES grafiku.

ZA UČENIKE SREDNJIH ŠKOLA

IRO „Građevinska knjiga", Beograd
Trg Marksa i Engelsa 8/II, tel. 347-662

NARUDŽBENICA — Računari u vašoj školi

Naručujem knjige pod rednim brojem ______

Ukupni iznos od ______ dinara uplatiću na žiro-račun IRO „Građevinska knjiga" br. 60801-603-15416
U slučaju spora nadležan je sud u Beogradu

Kupac ______

Adresa ______

Zaposlen ______

M. P.

______
Potpis kupca br. l. k., izdata od

Za iznose preko 30 000 moguće plaćanje u tri rate uz obaveznu overu RO

BAJTOVI U PLAVOM

Osnovna škola računarstva

Dejan Ristanović

Razvoj personalnih računara traje već dobrih desetak godina i to u smeru koji je malo ko predviđao: očekivalo bi se da različitih modela u početku bude malo a onda, iz godine u godinu, sve više. Početkom osamdesetih godina međutim, na tržištu je postojao džinovski broj međusobno nekompatibilnih mikroračunara koje su uglavnom koristili hobisti, dok su krajem osamdesetih softverske firme koje se bave „ozbiljnim" programima orijentisane jedva na tri mašine: IBM PC, mekintoš i atari ST. U jugoslovenskim uslovima dilema je još manja: postoji samo jedan računar koji se poslovno primenjuje — IBM PC. U tom bi se smislu moglo reći da poznavanje IBM PC-ja i njegove upotrebe predstavlja osnovnu školu računarstva. Ovaj celoviti prikaz posvećujemo korisnicima koji su za IBM PC možda ponekad čuli, ali koji o njemu ne znaju praktično ništa.

IBM PC se pojavio na tržištu tokom 1981. godine kada je gigantska multinacionalna kompanija IBM *(International Business Machines)* odlučila da udostoji svet jednim personalnim računarom — IBM je do tada proizvodio i veoma uspešno prodavao isključivo velike kompjuterske sisteme i prateću opremu. Sam računar konstruisala je relativno mala ekipa inženjera koja je, gledano sa istorijske distance od sedam godina, uradila izvanredan posao — računar je zamišljen toliko fleksibilno da su ga brojna proširenja održala na tržištu do današnjih dana, tj. daleko duže nego što su verovatno i najveći IBM-ovi optimisti očekivali. Što se izbora imena tiče, IBM-ovi stručnjaci za marketing nisu pokazali neku čudesnu imaginaciju — novi kompjuter je nazvan IBM PC odnosno IBM *Personal Computer*. Poseban je akcenat, dakle, stavljen na reč *Personal* — PC je računar namenjen **jednom** korisniku koji bi trebalo da zameni veće kompjuterske sisteme koji, istina, mogu istovremeno opsluživati više korisnika, ali koji su nesrazmerno skuplji i glomazniji. Ovakva se koncepcija potvrdila u praksi: IBM PC i njegovi naslednici u ovom trenutku predstavljaju *de facto* računarski standard poslovnih ljudi!

Šta čini IBM toliko popularnim? Pre svega fleksibilno i klasično dizajniran hardver koji se u ovom trenutku teško može označiti kao vrhunac razvoja elektronike, dobro testiran, klasičan (i opet prilično zastareo) operativni sistem i više od svega toga džinovska biblioteka programa.

Hardver

Hardver PC računara sastoji se od osnovne ploče sa mikroprocesorom, memorijom i raznim kontrolerima, izvora za napajanje, video adaptera, jedinica spoljne memorije, monitora i tastature. Bacimo kratak pogled na svaku od ovih komponenti.

Procesorska ploča

Procesorska ploča predstavlja suštinu računara — moglo bi se čak reći da sve ostale komponente sistema predstavljaju dodatak uz procesorsku ploču. Zavisno od modela kompjutera za koji se opredelimo (o modelima PC-ja govorićemo nešto kasnije), procesorska ploča može da bude deklarisana kao PC, XT ili AT, pri čemu su PC ploče praktično izumrle.

U centru svake ploče je mikroprocesor, integrisano kolo koje vrši kompletnu obradu podataka i koordinira radom ostalih komponenata računarskog sistema. Procesor je, jednostavno rečeno, deo računara koji može da se programira za procesiranje, tj. obradu nekakvih podataka. Svi IBM-ovi personalni računari zasnovani su na mikroprocesorima koje je projektovala i proizvela poznata firma Intel.



Neformalni svetski standard za poslovne računare: IBM PC/XT

Intelovi mikroprocesori

Tvorac prvog mikroprocesora je Intelov inženjer Marsijan Hof *(Marcian Hoff)* koji je 1970. godine došao na revolucionarnu ideju da projektuje integrisano kolo koje bi samostalno obavljalo određen deo posla procesorskih ploča koje su se ugrađivale u velike računare. Pokazalo se da i na tadašnjem stupnju razvoja tehnologije integrisano kolo može da preuzme kompletnu funkciju procesora i tako je nastao Intel 4004 koji je dobio oznaku po tome što je podatke obrađivao u grupama od po četiri bita: kažemo da je 4004 *četvorobitni* mikroprocesor.

Značajan komercijalni uspeh mikroprocesora 4004 učinio je da se Intel zainteresuje za ovakvu proizvodnju i da jedva godinu dana docnije ponudi tržištu mikroprocesor 8008, neposrednog osmobitnog naslednika 4004. Smatra se da 4004 i 8008 čine prvu generaciju mikroprocesora.

Na drugu generaciju nije trebalo dugo čekati — 1974. Intel predstavlja mikroprocesore 8080 i nešto docnije 8085. Premda i dalje osmobitni, ovi su mikroprocesori značajno moćniji od svojih prethodnika i predstavljaju prve mikroprocesore koji u potpunosti zaslužuje epitet „opšte namene". Karakteristike 8080 i naročito 8085 su

učinile da ovi čipovi mnogostruko nadžive svoj prirodni vek pa se i danas nalaze u mnogim računarima i specijelizovanim kontrolerima

Treća generacija Intelovih mikroprocesora ugledala je svet četiri godine posle druge — sredinom 1978 završen je razvoj mikroprocesora 8086 i 8088 8088 i 8086 su, sa programerske tačke gledišta praktično identični — programe pisane za jedan od njih bez ikakvih izmena može da izvršava drugi Razlike su dakle harverske — 8088 je jeftiniji mikroprocesor čije je povezivanje sa periferijom znatno lakše i jeftinije ali je zato saobraćaj sporiji

Prvi računari iz serije IBM PC zasnovani su na mikroprocesoru 8088 koji se kao što smo videli nalazi na prelazu između osmobitnih i šesnaestobitnih Neke docnije kopije IBM PC-ja zasnivane su na mikroprocesoru 8086 koji je jasno ubrzao izvršavanje programa ne ugrožavajući ni u kom smislu kompatibilnost. Mikroprocesor 8086 međutim ni u kom slučaju ne predstavlja vrh PC-linije sve ga češće zamenjuje neki od naslednika

Programeri nikada nisu imali naročito mišljenje o mikroprocesorima 8088 i 8086 najviše zamerki upućuje se njihovoj arhitekturi koja značajno otežava pristup kompletnoj raspoloživoj memoriji — šta vam vredi 640 kilobajta RAM-a kada bejzik ili paskal programi mogu da budu dugi najviše 64 kilobajta! Intel je na ove zamerke odgovorio 1984 predstavljajući četvrtu generaciju mikroprocesora čiji je glavni predstavnik 80286 Asemblersko programiranje mikroprocesora 80286 postalo je u mnogome slično programiranju na nekom višem jeziku uvedeni su složeni tipovi podataka kao što su stringovi ili racionalni brojevi i obezbeđena veoma komplikovana adresiranja koja olakšavaju pristup statičkim i dinamičkim matricama pointerskim strukturama i slogovima Arhitektura mikroprocesora 80286 je osim toga prilagođena modularnom programiranju i obezbeđuje zaštitu pojedinih segmenata memorije od neopreznog upisivanja podataka što je preduslov za projektovanje operativnih sistema koji obezbeđuju istovremeno izvršavanje većeg broja programa Mikroprocesor 80286 najzad može da adresira čitavih 16 megabajta memorije što bi trebalo da bude više nego dovoljno kako za postojeće aplikacije tako i za mnoge buduće potrebe korisnika računara.

Mikroprocesor 80286 je dokazao da svaki uspeh ima svoju cenu uspeh ranijih generacija Intelovih mikroprocesora počeo je da predstavlja ozbiljan balast! Pokazalo se naime da su dizajniranje mikroprocesora i hardvera personalnog računara važni za njegov tržišni uspeh, ali da se ključ pravog uspeha krije u softveru — novi računar ne treba opremiti samo operativnim sistemom i interpretatorima' odnosno kompajlerima raznih jezika, već i stotinama programa koji obezbeđuju raznorazne primene počev od univerzalno potrebne obrade teksta pa do vođenja složenih knjigovodstava koja interesuju veoma uzak krug korisnika

Intelovi projektanti nisu mogli da dopuste da džinovska softverska biblioteka razvijena za IBM PC i prema tome 8086 bude bačena pa su učinili mikroprocesor 80286 u potpunosti vertikalno kompatibilnim sa 8086 svaki program pisan za 8086 bez problema će se izvršiti i na 80286! Iako je ovakva odluka omogućila IBM-u da proda milione računara iz serije IBM PC AT (više o njima docnije) koji su zasnovani na mikroprocesoru 80286 i prema tome kompatibilni sa PC-jem pokazalo se da mikroprocesor 80286 u realnim situacijama uglavnom emulira svog starijeg brata i samim tim pati od njegovog 64-kilobajtnog limita Nove instrukcije koristi malo koji proizvođač softvera jer strogo usmeravanje nekog paketa prema IBM PC AT-u automatski umanjuje potencijalno tržište Zanimljivo je da se tek početkom ove godine na tržištu pojavio operativni sistem pisan specijalno za mikroprocesor 80286 — operativni sistem nazvan OS/2 ima veliku IBM-ovu podršku ali njegov tržišni uspeh time nije zagarantovan!

Odgovarajući na zamerke mnogih programera Intel je prošle godine predstavio petu generaciju svojih mikroprocesora čiji je predstavnik mikroprocesor 80386 U skladu sa pomenutom filozofijom vertikalne kompatibilnosti 80386 u sebi sadrži sve svoje prethodnike što znači da se programi pisani za IBM PC i IBM PC AT bez problema izvršavaju na računarima sagrađenim oko mikroprocesora 80386 Primenom jedne instrukcije mikroprocesor međutim može da aktivira sasvim novi set 32-bitnih instrukcija koje ukidaju memorijske segmente obezbeđuju zaštitu delova memorije i prema tome pravi višeprogramski rad složenu kontrolu pristupa memoriji i što nikako nije najnevažnije adresiraju preko 4 gigabajta ROM-a i RAM-a! Tržišni uspeh mikroprocesora 80386 unekoliko dovodi u pitanje jedino Intelove namere da u bliskoj budućnosti pripremi novu generaciju čiji će glavni predstavnik biti mikroprocesor 80486

Aritmetički koprocesor

Uspehu Intelovih mikroprocesora u mnogome su doprineli i prateći aritmetički koprocesori 8087 80287 i 80387 Svaki mikroprocesor se, naime može programski osposobiti za rad sa racionalnim brojevima i računanje vrednosti elementarnih funkcija ali je ovakav rad pogotovu na jednostavnijim mikroprocesorima uglavnom neprijatno spor Aritmetički koprocesor je zapravo specijalizovani mikroprocesor koji radi sa racionalnim brojevima — sve su operacije rešene na nivou mikrokoda pa se izvršavaju izuzetno brzo Iako su aritmetički koprocesori relativno nezgodni za masovnu proizvodnju i prema tome daleko skuplji od odgovarajućih mikroprocesora (8087 na primer ima tri puta više tranzistora nego 8086) ugradnja 8087 je mnogim vlasnicima PC-ja približila primene koje zahtevaju dosta računanje kao što su računarska grafika i kompjutersko projektovanje Vlasnici IBM PC AT-a moraju da nabave aritmetički koprocesor 80287, dok se 32-bitni računari zasnovani na 80386 dopunjavaju aritmetičkim koprocesorom 80387 Treba znati da aritmetički koprocesor praktično nikada nije integralni deo PC računara već ga treba odvojeno dokupiti što je pogotovu kada se radi o 80287 ili 80387 priličan izdatak

Frekvencija oscilatora

Kompletan rad računara odigrava se u taktu oscilatora tj kvarc kristala Viša frekvencija oscilatora automatski implicira brži rad kompjutera ali istovremeno zahteva ugradnju boljih tj skupljih komponenata Originalni IBM PC radi na 4 77 megaherca ali su se konstruktori većine današnjih PC računara opredelili za rad na 8 i 10 MHz ova varijanta je nazvana Turbo XT IBM PC AT računari obično rade na 10 ili 12 MHz dok je radna frekvencija mikroprocesora 80386 retko manja od 16 MHz

Neki PC računari su opremljeni prekidačem koji omogućava izbor jedne od radnih frekvencija (npr 4 77 6 10 ili 12 MHz) i tako obezbeđuje izvršavanje nekih veoma starih programa koji ne mogu da se prilagode bržem kloku Treba međutim znati da ubrzanje oscilatora teško može da spase loš program — spektakularni dobici u brzini postižu se isključivo promenom primenjenog algoritma!

Memorija

Memorija PC računara sastoji se od ROM-a i RAM-a ROM (*Read Only Memory* odnosno memorija koja može samo da se čita) sadrži 8 kilobajta programa koji stupa na scenu čim uključite računar — takozvani BIOS *(Basic Input Output System)* testira hardver i sa diskete učitava operativni sistem BIOS je aktivan i tokom rada računara — operativni sistem posredstvom njega komunicira sa tastaturom diskom i drugim komponentama sistema

Obzirom da IBM ne dopušta kopiranje BIOS-a, mnogi renomirani proizvođači kopija sastavili su funkcionalno ekvivalentne iako formalno sasvim drugačije varijante BIOS-a koje su verovatno ugrađene i u PC klona koji posedujete Većina ovih kopija zadovoljava zahteve svih bitnih komercijalnih programa pri čemu je rad često delimično ubrzan Ukoliko neki od alternativnih BIOS-a ne zadovolji vaše potrebe bez mnogo problema ga možete zameniti originalnim IBM-ovim ROM-om

U ROM bi u skladu sa IBM-ovim standardima trebao da bude ugrađen i bejzik interpretator Pokazalo se međutim da su zakoni o kopirajtu dovoljno snažni da ne omoguće proi-



zvođačima kopija da u PC ugrade ovaj deo ROM-a pa se većina klonova prodaje bez bejzika. Obezbeđena su naravno podnožja u koja mogu da se umetnu EPROM-i sa bejzikom ali ovakvoj ugradnji malo ko pribegava. EPROM-i (potrebno je osam EPROM-a 2764) nisu skupi njihovo kopiranje ne predstavlja poseban problem ali bejzik u ROM-u jednostavno nije potreban — program jednostavno učitavamo sa diska.

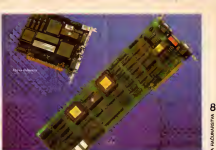
Priča o kapacitetu RAM-a (*Random Access Memory* ili memorija čiji se sadržaj može brzo menjati) je znatno duža. Prvi PC računari koje je IBM ponudio tržištu imali su samo 64 kilobajta RAM-a, ali je paralelno sa padom cena memorijskih čipova apetit korisnika rastao. Danas se smatra da je 256 kilobajta apsolutni minimum koji obezbeđuje normalno funkcionisanje IBM PC-ja i primenu važnih komercijalnih programa, većina PC-ja je međutim opremljena maksimalnim RAM-om koji mikroprocesori 8088 i 8086 mogu da adresiraju — 640 kilobajta. Ukoliko planirate kupovinu IBM PC AT-a najbolje je da se opredelite za megabajt RAM-a, dok apetiti mašina zasnovanih na 80386 još nisu definisani. U samom početku međutim morate da razumete i prihvatite jedno ograničenje: činjenica da ste se opremili RAM-om od 640 kilobajta ne mora da znači da ćete u jednom trenutku moći da sortirate 300 000 celih brojeva ili obrađujete 300 stranica teksta — neke „začkoljice" u arhitekturi mikroprocesora 8086 su učinile da memorije bude podeljena na segmente od po 64 kilobajta, većina komercijalnih programa zahteva da svi podaci stanu u jedan segment, što znači da se PC često degradira u osmobitnu mašinu.

Izvor za napajanje

Izvor za napajanje ili u stranoj literaturi PSU *(Power Supply Unit)* treba da zadovolji kako potrebe procesorske ploče tako i potrebe diskova, disk jedinica i raznih kartica za proširenja. Zato je čak i neka jeftina kopija PC-ja obično opremljena izvorom za napajanje od 130 vata, dok se moćniji AT računari prodaju sa izvorom od 200 vata. Karakteristika po kojoj se rangiraju izvori za napajanje je dakle kvalitet ugrađenog ventilatora — tiši ventilator obezbeđuje komforniji rad. Premda se neki PC klonovi prodaju i bez ventilatora, svakako bismo vam savetovali da se ne lišavate pomoći ove naprave.

Video adapter i monitor

Glavni problem pri konfigurisanju PC računara svakako je izbor video adaptera i monitora — treba se opredeliti između kartica koje nose zagonetna imena CGA, EGA, VGA i Herkules. Za čitavu zbrku kriv je ponajviše IBM koji je PC računare koncipirao ne vodeći nikakvog računa o grafici. Razlika za ovu nemarnost nije teško pogoditi: IBM je godinama proizvodio velike računarske sisteme i smatrao da je stekao vrlo dobar utisak o tome kakvi su terminali potrebni korisnicima — posetite bilo koji računski centar i pronaći ćete terminale poput VT 50, VT 100 ili u boljem slučaju VT 200. Grafičke mogućnosti ovih terminala su, verovali ili ne, gotovo nikakve — ekran je monohromski, tj crno-beli ili zeleno-beli, radi se isključivo sa slovima i (eventualno) grafičkim karakterima, skrolovanje teksta nije baš impresivno brzo... Razlog za ovakve karakteristike nije samo težnja da se cena snizi — racionalni rad velikog sistema zahteva da se na komunikaciju sa terminalima troši što manje procesorskih sekundi, dok rad sa grafikom zahteva prenos ogromnih blokova podataka (da li je lakše poslati ASCII kod slova „A" ili njegovu bit mapu?). Samo se po sebi razume da je grafika neophodna za neke specijalne aplikacije ali korisnici kojima su ovakve aplikacije potrebne rado investiraju prilična sredstva u grafičke terminale, laserske štampače, plotere i sličnu opremu.



Grafičke kartice najnovije generacije. Fino crtanje (1280×600 tačaka) u 256 boja iz palete od 256.000

Pošto je PC zamišljen kao jeftin poslovni sistem, IBM-ovi inženjeri su smatrali da on može sasvim lepo da živi bez grafike — na raznim velikim računarima su na kraju krajeva implementirane sve moguće poslovne aplikacije, pa ipak ni jedan korisnik nije zakukao za grafikom! IBM se najzad potrudio da skup znakova bude začuđujuće kompletan i da obuhvati kako slova raznih stranih azbuka (nemačka, švedska itd) tako i razne matematičke simbole i neophodna grčka slova. Setom karaktera je uz to obezbeđena i nekakva rudimentarna grafika — jednostruke i dvostruke linije i svi njihovi neophodni preseci. Prvi poslovni programi su zaista funkcionisali bez grafike. A onda su se pojavile prve grafičke kartice, pa su autori softvera počeli da dopunjavaju svoja remek dela okvirima, linijama, osenčenim površinama i sličnim specijalitetima. Grafika je u početku bila samo dekorativna, a onda je paket Lotus izazvao pravu malu revoluciju — po prvi put je personalni računar upoznao integrisani program za unakrsna izračunavanja i poslovnu grafiku.

Najrazličitije firme su jednostavno poludele za Lotusom, što je značilo da stotine njihovih kompjutera treba dopuniti grafičkom karticom — pitanje je samo kojom. Najprirodnije je obratiti se IBM-u, ali je ovakvo obraćanje davalo sumnjive rezultate — IBM je prodavao samo glupost zvanu CGA. CGA kartica, istini za volju, otvara PC-ju grafička perspektive (grafička rezolucija 640•200 i četiri boje iz palete od 16) ali njena ugradnja podrazumeva prilično ružna slova, nisku rezoluciju, spor rad i relativno ružne slike. Možda bi sve to nekako i prošlo, CGA kartica međutim zahteva da kupite nov kolor monitor, što znači da stari (skup ali izvanredan) IBM-ov monohrom monitor možete da bacite. Mala nezavisna firma Herkules povukla je mnogo bolji potez — kada kupite Herkules karticu, na starom IBM-ovom monohrom monitoru imaćete lepa slova i sasvim pristojnu crno-belu grafiku 720•346! Mnoge su se firme odlučile za ovakvu nabavku, pa je Herkules kartica postala gotovo oficijelni standard, svi važniji komercijalni programi kompatibilni su sa ovim adapterom. Neki stariji programi, igre i naročito bejzik interpretatori nisu potpuno kompatibilni sa Herkulesom, što za početnika predstavlja veoma ozbiljan problem.

Herkules kartica je, rekosmo, u svetu prilično zastupljena ali je u Jugoslaviji njena zastupljenost praktično stoprocentna — autor ovoga teksta koji ima (ne)sreću da koristi EGA karticu je morao da preinstalira svaki program koji mu je došao do ruke! Da stvar bude još gora, čitava domaća softverska produkcija je zasnovana na Herkules kartici, što proizvodi mnogo komičnih efekata: razni tekst editori na EGA-i rade savršeno, ali se na ekranu jednostavno ne vidi ništa — ako znate da je treći znak pete linije pogrešan, možete da ga promenite ali ovakav rad teško može da se preporuči. Da bi neki program sarađivao sa raznim karticama, treba se uzdržati od direktnog upisivanja teksta u video memoriju i koristiti usluge BIOS-a ili operativnog sistema.

Umesto Herkulesa ili već pomenute CGA kartice možete da se opredelite za neku noviju IBM-ovu karticu kao što je EGA *(Enhanced Graphic Adapter)*. EGA je IBM-ova poboljšana grafička kartica koja nudi izvanredan ispis teksta, grafičku rezoluciju 640•350 tačaka i izbor 16 boja iz palete od 64, ali koja zato nije jeftina — kupovina EGA kartice podrazumeva i kupovinu kvalitetnog EGA kompatibilnog kolor monitora. VGA je najzad kartica koja se ugrađuje u nove računare iz IBM-ove serije PS/2 — namenjena je korisnicima koji tragaju za većom grafičkom rezolucijom, većim brojem boja, bržim radom i naravno gotovo astronomskom cenom. Što se karakteristika tiče, VGA emulira EGA i CGA standarde, a ipak omogućava i daleko zanimljivije modove: 640•480 u 256 boja koje se biraju iz fantastične palete od 262144! Ljudima koji se profesionalno bave grafikom i animacijom je na raspolaganju specijalni mod sa rezolucijom 1024•768 u 256 boja.

Nevolja sa karticama je što svaka zahteva svoj monitor: uz IBM-ovu MDA karticu (standardni crno-beli ispis teksta bez grafike) i Herkules treba kupiti takozvani TTL monitor. CGA kartica zahteva monitor sa takozvanim kompozitnim video ulazom. EGA i VGA zahtevaju RGB kolor monitore, s tim što se frekvencije horizontalnog i vertikalnog skaniranja ovih monitora razlikuju.
U poslednje vreme sve su popularniji takozvani multisinhroni monitori koji su, istini za volju, skuplji od standardnih, ali koji se automatski

prilagođavaju bilo kojoj postojećoj i mnogim budućim grafičkim karticama. Ukoliko, dakle, planirate kupovinu kolor monitora, slobodno se opredelite za multisinhroni, jer na taj način investirate u sadašnjost i budućnost

Tastatura

Pri radu sa računarom korisnik praktično nema prilike da ustanovi da li je poreklo centralne kutije tajvansko ili IBM-ovo, ali se vrednost tastature odmah oseti. Originalni IBM PC, IBM PC AT i IBM PS/2 su snabdeveni izvanrednim (mehaničkim) tastaturama koje se sa centralnom jedinicom povezuju relativno dugim kablom — treba znati da se PC tastature ne može upotrebiti na AT-u i PS/2 i obratno. Uz razne klonove se isporučuju razne na prvi pogled identične tastature veoma promenljivog kvaliteta i cene. Investicija u bolju (npr. originalnu) tastaturu i kvalitetan monitor je vrlo isplativa — ova dva uređaja praktično predstavljaju jedine komponente kompjuterskog sistema sa kojima neposredno komuniciramo!

Spoljna memorija

Izbor medija spoljne memorije predstavlja još jednu zagonetku: možete da kupite disk jedinice od 360 kilobajta ili 1.2 megabajta i hard diskove raznih kapaciteta. Što se hard diskova tiče, većina Jugoslovena se odlučuje za umerene diskove od 20 megabajta, premda poneko investira i u daleko skuplje jedinice od 40 ili 60 megabajta — standardni operativni sistem se ne snalazi sa diskovima većim od 32 M ali se da očekivati da će ovo ograničenje biti brzo prevaziđeno.

Izbor flopi disk jedinice je mnogo veći problem. Možete pre svega da nabavite disk jedinicu od 5.25 ili 3.5 inča — prva predstavlja standard prošlosti, a druga stvar budućnosti, pri čemu je u ovom trenutku prošlost primamljivija: 99% vlasnika PC-ja poseduje disk jedinice od 5.25 inča, što znači da vlasnici novijih disketa, ako nisu raspoloženi za kupovinu originalnog softvera, u Jugoslaviji teško dolaze do programa. Slična se logika unekoliko prostire i na disk jedinice od 1.2 megabajta: na njoj snimljene diskete su često nečitljive na običnom PC-ju, što otežava razmenu programa i podataka. Većina korisnika, dakle, nabavlja jednu disk jedinicu od 360 kilobajta i hard disk od 20 megabajta — smatra se da je ovako konfigurisan računar idealan za razne primene. Ukoliko su novčani fondovi tanji, treba se opredeliti za PC sa dva flopi diska od po 360 kilobajta — jedna disk jedinica nije dovoljna za iole ozbiljne poslovne primene. Ovaj zaključak je na prvi pogled prilično čudan: ljudi obrađuju tekst i na računarima koji opšte sa kasetofonom! Iako bi jedna disk jedinica, propraćena velikom PC-jevom memorijom, teorijski bila sasvim dovoljna za ovakvu primenu, autori brojnih tekst procesora pretpostavili su da svaki korisnik ima hard disk i razvijali programe ne obraćajući pažnju na njegovu dužinu!

Poslednjih godina sve veći deo građanskih prava stiču strimer trake od 20 ili 40 megabajta koje obezbeđuju brzo i automatizovano kopiranje kompletnog sadržaja hard diska na relativno jeftinu magnetnu traku. Ovakve se kopije pripremaju da bi se minimizovala šteta koja nastaje kada, usled neizbežnog defekta opreme ili programerske greške, podaci na disku budu uništeni ili oštećeni.

Proširenja

Jedna od osnovnih karakteristika PC računara je izuzetna otvorenost — svaki PC ima određen broj takozvanih slotova u koja se umeću razne kartice. Tri slota svakog PC-a su obično popunjena grafičkim adapterom, kontrolerom



Maksimalna fleksibilnost u komponovanju sistema. Kartice za XT i AT modele

disk jedinica i kontrolerom za hard disk. Četvrti slot rezervišemo za takozvanu multifunkcijsku karticu koja predstavlja kombinaciju paralelnog interfejsa za štampač (Centronics port), serijskog interfejsa (RS-232) koji obezbeđuje priključenje miša, modema ili serijskog štampača i baterijski napajani časovnik koji „pamti" vreme i datum čak i dok je računar isključen i i u saradnji sa operativnim sistemom, olakšava snalaženje sa raznim verzijama istih programa, tekstova ili podataka. Obzirom da na mnogim IBM PC AT kompatibilnim računarima časovnik realnog vremena predstavlja integralni deo procesorske ploče i da su mnoge grafička kartice (npr. Herkules) opremljene paralelnim interfejsima za štampač, multifunkcijska kartica se često svodi na jednostavan RS-232 interfejs.

Preostali ekspanzioni slotovi (ima ih obično 1—3) obezbeđuju priključenje raznih kartica koje su vam potrebne za posao kojim se bavite — to mogu da budu kontroleri, komunikacioni adapteri, AD i DA konvertori i tome slično. Posebnu pažnju treba obratiti na memorijska proširenja — iako smo videli da IBM PC i IBM PC XT mogu da adresiraju najviše 640 kilobajta RAM-a, neke nezavisne firme proizvode kartice sa dva, četiri ili čak osam megabajta RAM-a, ovakav RAM može da se koristi samo na neke specijalne načine i u okviru nekih aplikacionih programa, što znači da je upotrebljivost memorijskih proširenja prilično sumnjiva.

PC svakako treba dopuniti i veoma jeftinim dodatkom koji se zove miš *(mouse)*. On ubrzava komunikaciju sa raznim komercijalnim programima i predstavlja neophodan preduslov za računarsko crtanje.

Modeli PC računara

Pod imenom IBM PC podrazumeva se mnogo različitih računara: tu je, pre svega, originalni IBM PC sa malo memorije i malo portova za

ekspanziju koji se više ne proizvodi. Tu je, zatim, IBM PC XT koji koristi najveći broj vlasnika personalnih računara — kada se, zapravo, kaže IBM PC, misli se na IBM PC XT. IBM, razume se, nije stao samo na tome. Najpre se pojavio model AT, a potom i potpuno nova serija PS/2.

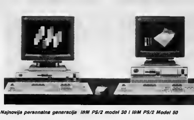
Korak dalje napravio je IBM PC AT (AT dolazi od *Advanced Technology)* koji je zasnovan na šesnaestobitnom mikroprocesoru 80286 ali on uglavnom emulira svog starijeg brata 8086. Možda ćete se začuditi kada pročitate da je IBM PC AT (od sada ćemo ga radi malo kraćeg pisanja zvati samo AT) opremljen sa 256 kilobajta RAM-a čak i najobičniji PC kompatibilan tajvanac obično ima čitavih 640 K. IBM je međutim oduvek pomalo škrtario na memoriji (nešto slično rade i druge velike firme kao što je Hewlett Packard) — u njegove se računare ugrađuju izuzetno kvalitetni dinamički RAM čipovi sa automatskom proverom parnosti — takve su memorije bez obzira na pad cena i dalje relativno skupe. IBM osim toga računa da će dobar deo kupaca povećavati memoriju i da će se sasvim prirodno obratiti proizvođaču računara. to je izvanredan izvor prihoda i dobro poznata politika velikih firmi.

IBM PC AT

Ukoliko IBM ubira novac od prodaje memorijskih proširenja AT je predodređen da donese ogromne sume. memorija se na osnovnoj ploči proširuje do 512 kilobajta a preko slotova za čak i do 16 megabajta! Za ovako veliki adresni prostor je zaslužen uglavnom novi mikroprocesor 80286 koji je istovremeno i glavni krivac što ovako veliki RAM ostaje prilično slabo iskorišćen — ni AT nije imun na segmentiranje memorije. Promenom mikroprocesora je međutim mnogo dobijeno i na drugim stranama. najvažnije je ubrzavanje rada koje ne potiče toliko od kloka (6 MHz prema PC-jevih 4.77) i šesnaestobitne magistrale za podatke koliko od unapređene arhitekture. Izvršavanje većine instrukcija, naime, zahteva dvostruko manje otkucaja kloka, obzirom da se faza pripreme neke instrukcije preklapa (tj. obavlja istovremeno) sa fazom izvršenja prethodne. Sumarni efekat je dva do četiri puta brže izvođenje svih komercijalnih programa.

Moćniji računar obično zahteva i odlju tastaturu pa je IBM kompletno predizajnirao i ranije skoro savršenu tastaturu težeći da je učini što kvalitetnijom i pogodnijom za dugotrajnu upotrebu. Razlike se uočavaju na prvi pogled. *Shift, Control, Backspace* i *ENTER* su povećani a ređe korišćene funkcije i znaci (npr. obrnuta kosa crta, obrnuti apostrof, *print screen* i slični) pomerani prema periferiji. Jedini novi taster *Sys Req* šalje interapt mikroprocesoru i tako skreće njegovu pažnju na nešto što korisnik trenutno preduzima. Akcije koje će računar preduzeti po prijemu ovog interapta zavise od operativnog sistema. za sada se on ignoriše premda jasno može da bude upotrebljen u nekom komercijalnom programu. U oči najzad upadaju tri LED diode koje su mnogo značajnije nego što mislite da bi se one uvele. tastatura je morala da prestane da bude ekskluzivni otpremnik. Kao što tastatura PC-ja šalje kodove računaru, tako i AT može da šalje naredbe novoj tastaturi. neke od njih naredbi su paljenje *Caps Lock, Scroll Lock* i *Num Lock* indikatora.

Neposredna posledica svih ovih promena je nekompatibilnost. tastaturu PC-ja ne možete da prikačite na AT niti tastaturu AT-a na PC ili XT.



Najnovija personalna generacija IBM PS/2 model 30 i IBM PS/2 Model 50

Ukoliko ipak razmišljate o sličnim avanturama dopustite da vas upozorimo. razlike nisu samo softverska što znači da bi bila neophodna i gradnja nekog manje ili više složenog interfejsa.

AT je opremljen prilično moćnim napajanjem koje može da povuče 190 vati (uporedite to sa 63 W na PC-ju i 130 W na XT-u) tu je naravno i neizbežni ventilator. Jači ispravljač omogućava priključenje većeg broja kartica (obezbeđeno je osam slotova) pri čemu su posebno privlačni hard diskovi od 20 ili 40 megabajta. U osnovnu je konfiguraciju jasno uključena i jedna disk jedinica koja omogućava upisivanje čak 1.2 megabajta podataka na standardnu disketu. Podaci se upisuju na obe strane diskete i to na po 80 traka svaka traka ima 15 sektora po 512 bajta. Samo se po sebi razume da ove disk jedinice mogu da čitaju ali ne i da pouzdano pišu po starim disketama od 360 K.

Ovaj kratki prikaz At-a završavamo jednom ne mnogo važnom pikanterijom. AT je jedan od retkih personalaca koji mogu da se zaključaju! Ne mislimo naravno na zaključavanje u orman (to uz vašu pomoć ume i spektrum) niti na paljenje pomoću ključa. operativni sistem može da testira da li je specijalni ključ okrenut i u zavisnosti od toga prima ili ignoriše naredbe sa tastature. Primena koja prva pada na pamet dolazi iz sfere bankarstva. svaki službenik može da pročita stanje tekućeg računa, ali samo onaj ko ima ključ sme da registruje neku uplatu ili isplatu.

PS/2

Prvo tromesečje 1987. donelo nam je IBM-ovu seriju računara nazvanu IBM PS/2 odnosno *IBM Personal System/2*. ime treba da asocira na sistem koji je još više približen korisniku a ipak sličan prethodnim generacijama. Kompatibilnost sa ranijim modelima je filozofija na kojoj IBM već više od pola veka zasniva svoje velike uspehe pa je tako i serija PS/2 u velikoj meri kompatibilna sa starim dobrim PC-jem i AT-om.

PS/2 serije se sastoji od pet modela koji su dobili oznake 25, 30, 50, 60 i 80. Modeli 25 i 30 su poboljšani IBM PC XT tj. računari zasnovani na mikroprocesoru 8086 modeli 50 i 60 su verzija IBM PC AT-a zasnovane na 80286 dok se u centru modela 80 nalazi trideset dvobitni Intelov mikroprocesor 80386. Svaki od modela predstavlja potpuno zaokružen sistem sa tastaturom, grafičkom karticom, masovnom memorijom, časovnikom realnog vremena, serijskim i paralelnim interfejsom, slotovima za ekspanziju i (opcionim) monitorom i to po ceni koja je vrlo pristupačna zapadnjačkom džepu (što se jugoslovenskog džepa tiče, možda ćemo morati da pričekamo još koju godinu) — najslabiji model košta 1100 a najjači 7000 funti! Da bi korisnik znao u šta ulaže tih 7000 funti, modeli 60 i 80 su čitavi ormančići koji stoje pored radnog stola!

Posebna pažnja je ponovo posvećena tastaturi koja je dodatno predizajnirana — dodato je nekoliko tastera dok je kvalitet izrade dalje poodljšan. Kontroli računara pomaže i miš koji se povezuje sa centralnom kutijom a ne kao kod Olivetijevih modela sa samom tastaturom.

Najznačajnije izmene pretrpeo je hardver — umesto za standardne, IBM se opredelio za specijalno dizajnirane čipove koji treba da uspora i otežaju neovlašćeno kopiranje. Naročita pažnja posvećena je grafici (već smo govorili o blagodetima VGA kartice) i *Micro Channel-u* u novoj i veoma brzoj internoj magistrali koja bitno unapređuje mehanizme protoka podataka. Radi se zapravo o 32-bitnoj magistrali koja se po potrebi „sužava" na 16 linija i koja je dopunjena hardverom koji obezbeđuje kontrolu prioriteta. Svakoj komponenti (u komponente ovde ubrajamo i mikroprocesor, DMA kontroler i slične „inteligentne" čipove) priključenoj na magistralu se dodeljuje prioritet koji u toku rada može da se menja po potrebi — ako jednom dokupite pločicu sa mikroprocesorom 80486 i dodatnim RAM-om, dodelićete joj visoki prioritet dok će se osnovni procesor baviti samo jednostavnijim poslovima i pristupati hardveru samo kad ovaj nije neophodan „glavnom gazdi".

Kada već pominjemo RAM, IBM ovoga puta nije škrtario na memoriji. model 30 je opremljen sa 640 K a ostali najmanje megabajtom. RAM modela 50, 60 i 80 može da se proširi do 16 megabajta pri čemu bi procesor 80386 mogao da adresira i čitava četiri gigabajta. ROM sadrži novi BIOS, specijalni ABIOS na kome će se zasnivati novi operativni sistem OS/2 i tradicionalni bejzik interpretator od koga proizvođači klonova obično odustaju. BIOS je jasno vertikalno kompatibilan sa ranijim verzijama što znači da će svi propisno pisani programi za PC, XT i AT raditi i na računarima iz serije PS/2. Problemi mogu da se očekuju jedino kod nekih igara koje direktno pristupaju hardveru ali — svi i onako znamo da igre na PC-ju nisu ni za šta!

Serija PS/2 donosi mnogobrojne novitete na polju masovne memorije. IBM je pre svega konačno odustao od „antičkog" standarda koji omogućava upisivanje svega 360 kilobajta informacija na disketu od 5.25 inča. primenjene su moderne diskete od 3.5 inča i novi format upisa od 720 kilobajta, odnosno 1.44 megabajta. Za jugoslovenske vlasnike PC računara ova promena nije naročito dobrodošla jer otežava razmenu programa i podataka — svaki PS/2 računar treba dopuniti i jednom standardnom disk jedinicom koja zahteva i odgovarajući kontroler.

IBM-ova serija PS/2 predstavlja punu afirmaciju tvrdnje da hard disk više nije egzotična naprava neophodna samo uskom krugu korisnika — svi modeli (osim 30—002) su opremljeni

masivnim diskovima od 20, 44, 70 ili čak 115 megabajta. Ovoliki kapaciteti će biti neophodni novom operativnom sistemu koji je nazvan OS/2 — hard diskovi na AT u su doduše mogli da se razdele na particije koje će biti posvećene raznim operativnim sistemima, ali je ova mogućnost retko korišćena. 99% PC kompatibilnih računara radi pod nekom od verzija MS DOS-a. Očekujući da će OS/2 računari raditi pod MS DOS-om, OS/2 i Unix-om, IBM se opredelio za diskove ogromnih kapaciteta koji, jasno, ne deluju baš blagotvorno na cenu. Kvalitetni diskovi uz to predstavljaju veliki problem za male tajvanske firme koje se za svoj deo PC kolača bore uglavnom bagatelnim cenama.

Priča o klonovima

Apsolutna dominacija jednog proizvođača računara na poslovnom tržištu, što se korisnika tiče, ima dobrih i loših strana. Dobra strana se svode na obilje raspoloživog softvera, proširenja i jednostavnu razmenu podataka sa saradnicima, a loše na izvesno gušenje kreativnog razvoja računarstva koje vlasnik monopola nužno izaziva.

Što se proizvođača računara tiče, dominacija „velikog plavog" nije baš pozdravljena — IBM je prigrabio dobar deo njihovog kolača i mnoge odveo pravo u bankrotstvo! Na svu sreću, u Sjedinjenim Državama postoji jedna (kod nas ideološki neprihvatljive) izreka koja glasi: *ako ne možeš da ih pobediš, ti im se pridruži*. Na kraju krajeva, ako proizvedete mašinu koja radi sve što i IBM-ov PC i još košta neku hiljadu dolara manje, napravićete divan posao. Compaq se prvi dosetio ove majstorije i tako je nastao prvi PC klon. Compaq je, uzgred budi rečeno, i dalje vrlo aktivan pa je pre IBM-a (!) ponudio tržišlu računar zasnovan na mikroprocesoru 80386 — DeskPro 386 se uspešno takmiči sa IBM-ovim modelom 80 iz serije PS/2.

Uzrečica koju smo pomenuli u prethodnom pasusu važi, izgleda, i u nekim evropskim zemljama. Čuveni Olivetti je napravio izvanredan posao konstruisavši M-24 i M-28 PC i AT klonove koji i danas predstavljaju jednu od okosnica razvoja italijanskog giganta. Jedan po jedan i ostali su se proizvođači uključivali u trku, dizajnirajući i proizvodeći PC klonove sa različitim uspehom. Tu su, najzad, i korejanci i tajvanci koji su se uvek pridruživali uspešnima ne gubeći vreme u pokušavanju da ih pobede.

Konstrukcija PC-ja na neki način podstiče masovno kopiranje — računar je zasnovan na standardnoj i višestruko proverenoj tehnologiji, nema specijalnih čipova i predstavlja školski primer otvorene mašine. Želeći da proširi tržišta, IBM je publikovao svu postojeću hardversku i softversku dokumentaciju! Cena klonova je neprekidno padala, da bi se ovih dana XT klon nepoznatog proizvođača nabavljao za samo 800–1000 maraka, što je desetak puta manje novca nego što su koštali primerci originala!

Memorija klonova je retko manja od 256 kilobajta (originalni IBM je, da podsetimo, bio opremljen RAM-om od svega 64 K), a uglavnom dostiže teorijski maksimum koji MS DOS podržava — 640 kilobajta. Tu je obično i crno-bela ili kolor video kartica koja mora da bude kompatibilna sa softverom koji je pisan za IBM PC, serijski interfejs i interfejs za (IBM kompatibilan) štampač, priključak za modem.

Obzirom na probleme sa kopirajtom BIOS-a, ni jedan klon koji legalno egzistira na tržištu nije i ne može da bude 100% IBM kompatibilan (ni sam IBM nije 100% IBM kompatibilan, jer je BIOS modela PS/2 značajno promenjen!). Nedostatak apsolutne kompatibilnosti ne mora da diskvalifikuje klon: ako računar izvršava sve programe koji su vama potrebni, nije vas mnogo briga što bi se možda mogla napisati rutina koja će ga



Najbrži klon na svetu. U Compaq Desk-Pro ugrađen je procesor 80386 na 20 MHz

zbuniti! Pri kupovini nekog od klonova obično se isprobava čuveni paket *Lotus 1-2-3* (neka od novih verzija sa specifičnim zaštitama), *WordPerfect*, *dBASE III Plus*, *SideKick*, *SuperKey* i, verovali ili ne, jedna čuvena igra: *Flight Simulator* (kada smo već kod igara, *Flight Simulator* je jedna od retkih dobrih igara za PC-ja).

Pri kupovini klonova možete se, dakle, opredeliti za proizvod renomiranog proizvođača ili za tajvansku kopiju. Argumenti u prilog kupovine proizvoda renomirane firme nisu nepoznati — dobijate mašinu u koju su ugrađene kvalitetne komponente (od tastatura pa do čipova), imate garanciju i obezbeđen servis, predviđenu ekspanziju... Dobre strane tajvanaca su niska cena i potpuno kloniranje koje ignoriše zakone o kopirajtu, a loše nedostatak bilo kakve garancije ili podrške proizvođača. I pored toga, većina Jugoslovena se bez razmišljanja opredeljuje za tajvanske kopije — neko i nastrada, ali je većina kupaca više nego zadovoljna!

Sistemski softver

Da biste bili u prilici da primenite potencijale vašeg PC--ja, morate da savladate relativno složenu metodsku jedinicu koja se zove PC DOS, odnosno MS DOS — kompletna komunikacija sa računarom svodi se na „razgovor" sa operativnim sistemom. Neće vam, na sreću, biti neophodno čitanje dugih DOS priručnika — informacije izložene u ovom poglavlju uglavnom su dovoljne za osnovne poslovne primene.

Pre početka rada računar treba sklopiti i konfigurisati, što ponekad nije tako lak posao — treba ugraditi razne kartice, postaviti mikroprekidače, formatirati hard disk, prepisati na njega odgovarajuće datoteke i tome slično. Svakom početniku bismo preporučili da ovu operaciju poveri nekom ko je malo iskusniji ili da, ako je baš ostavljen samom sebi, detaljno pročita uputstvo, neku knjigu o PC-ju ili početna poglavlja našeg umetka „MS DOS za početnike" iz „Računara 31". U daljem tekstu pretpostavićemo da je IBM PC koji koristite ispravno sklopljen i konfigurisan.

Razgovor sa DOS-om

Po uključivanju i ispisivanju odgovarajućih poruka, računar ispisuje takozvani prompt koji glasi A> ili C>. Prompt označava da je računar spreman da primi vašu komandu — otkucajte je i **obavezno** pritisnite ENTER (taster koji mi zovemo ENTER se na vašoj tastaturi možda zove RETURN ili je na njemu nacrtana samo neka čudna pravougaona strelica; uvek ćete ga naći sa desne strane tastature) stavljajući tako računaru do znanja da je on na potezu. Pritisak na ENTER u daljem tekstu nećemo posebno naglašavati — ako ga zaboravite, računar će satima strpljivo čekati da završite započetu naredbu.

Kakve naredbe ima smisla kucati? Sam DOS prepoznaje određen broj takozvanih *internih* naredbi koje smo pobrojali na slici 1. Tu su i programi koje ste prekopirali na radnu disketu ili disk — ti programi omogućavaju formatiranje, kopiranje, proveru i mnoge druge operacije sa diskovima. Programe sa DOS diskete pobrojali smo na slici 2, oni se zovu *eksterne* komande jer računar mora da učita njihov radni deo sa diska da bi ih izvršio.

slika 1

BREAK	DATE	PATH	TIME
COPY	DIR	PROMPT	TYPE
CHDIR	DEL	RENAME	VER
CLS	ERASE	RMDIR	VERIFY
CTTY	MKDIR	SET	VOL

slika 2

ASSIGN	COMP	FIND	MODE
SELLECT	AT-FORMAT	COPY	FORMAT
MORE	SETUP	AT-SETUP	DISKCOMP
GRAFTABL	PARK	SHARE	ATTRIB
DISKCOPY	GRAPHICS	PRINT	SORT
BACKUP	EXE2BIN	JOIN	RECOVER
SYS	CHKDSK	FDISK	LABEL
RESTORE	TREE		

Osim izvršavanja eksternih i internih komandi, MS DOS omogućava direktno startovanje programa. Opremili ste se, na primer, tekst procesorom koji se zove WP *(od WordPerfect)*. Da biste počeli da obrađujete tekst, otkucaćete WP, pritisnuti ENTER i — pročitati pozdravnu poruku. Program Lotus 1–2–3 na sličan način startujete otkucavši 123, dBASE III Plus sa DBASE i tako dalje. Samo je po sebi jasno da sve ove programe ne možete da smestite na jednu disketu — da biste startovali *WordPerfect* moraćete da ubacite disketu na koju je on snimljen i da tek onda otkucate WP.

Diskete se, kao što se sećamo, umeću u drajvove kojih može biti nekoliko. Ukoliko posedujete samo jednu disk jedinicu, ona se zove A. Ukoliko posedujete dve disk jedinice, zovu se A i B. Ukoliko, najzad, posedujete hard disk, njegovo je ime C, dok se disk jedinice zovu A i B (ako posedujete samo jednu disk jedinicu i hard disk, imena će respektivno biti A i C, jedinica B u ovom slučaju ne postoji). Kada otkucate neku komandu, računar pokušava da je pronađe u spisku internih i eksternih. Ukoliko ne uspe (mehanizam nije baš sasvim ovakav, ali je sličan je), pokušava da izvrši program sa diskete umetnute u *tekuću* disk jedinicu. Kada uključite računar, tekuća je disk jedinica A odnosno C — o tome vas obaveštava prompt. Ukoliko je program koji hoćemo da startujemo na disku B, otkucaćemo B: (ne zaboravite dvotačku!) i pritisnuti ENTER. Prompt se menja — na početku sledećeg reda piše B> što znači da je disk B sada tekući. Na disk A možemo da se vratimo kucajući A:, dok se hard disku obraćamo sa C:



Šta da radimo ako posedujamo samo jednu disk jedinicu? Pre nego što startujemo neki program izvadićemo DOS disketu i zameniti je disketom sa programom a zatim otkucati WP DBASE ili nešto drugo Kada se izvršavanje programa (regularno ili neregularno) završi računar će možda zahtevati da vratimo DOS disketu kako bi nastavio sa normalnim radom

U početku će nam se često događati da pogrešno otkucamo ime nekog programa ili da zaboravimo da umetnemo potrebnu disketu Računar će nas tada pozdraviti porukom *Bad command or file name* i sačekati novu (ispravnu) komandu Ukoliko sa druge strane disketa nije umetnuta ili ukoliko vrata drajva nisu zatvorena pojaviće se poruka poput *Drive not ready* a zatim pitanje *Abort Retry Ignore?* Pritisak na A prekida zahtevanu operaciju i ispisuje prompt pritisak na R ponavlja operaciju (pretpostavlja se da ste u međuvremenu otklonili smetnju npr umetnuli disketu) dok pritisak na I nalaže operativnom sistemu da ignoriše grešku i nastavi da izvršava program koji je zahtevao „nemoguću" operaciju iako se opcija A i I obično svode na isto predlažemo vam da u početku ne ignorišete greške — uvek pokušajte da nađete uzrok pa ga onda otklonite

Tajna PC tastature

Obzirom da sa PC-jem komunicirate posredstvom tastature vrlo je važno da naučite da koristite sve njene mogućnosti Otkucati naredbu i pritisnuti ENTER nije teško Pa ipak tastature treba da vam omogući i bezbolno ispravljanje grešaka i upotrebu mnogih komercijalnih programa Zato ćemo se ovde pozabaviti funkcijama svih specijalnih tastera PC-ja i AT-a — predlažemo vam da tekst pročitate i garantujemo da ćete mu se docnije s vremena na vreme vraćati

Trebalo bi pre svega da identifikujemo grupe tastera U centru su slova a iznad njih brojevi i specijalni znaci — baš kao na pisaćoj mašini Levo od slova (ili na nekim tastaturama iznad njih) su funkcijski tasteri kojih ima 10 i koji su obeleženi sa F1 F2 ... F10 Ove tastere uglavnom koriste komercijalni programi poput tekst procesora za razne stvari koje su detaljno opisane u okviru uputstva koje (ponekad) dobijate uz program Što se DOS-a tiče naučićemo da koristimo nekoliko funkcijskih tastera u toku ispravljanja komandnih linija

Sa desne strane se nalazi numerička tastatura koja se na PC-ju sastoji od 15 a na AT-u od 18 tastera Na ovim dirkama ispisani su brojevi koji unekoliko konkurišu gornjem redu tastera — pri unošenju veće količine podataka često je pogodnije koristiti numeričku tastaturu nego slediti logiku obične pisaće mašine U normalnim uslovima numerički tasteri se međutim koriste za kontrolu računara primetimo da su na njima iscrtane strelice odnosno napisane reči poput *Home End Insert PgUp* itd

Neke ne-IBM-ove tastature imaju još dve grupe tastera koji su obično smešteni između slova i numeričke tastature Prva grupa se sastoji od 6 tastera na kojima je ispisano *Home End PgUp PgDn Insert* i *Delete* a druge od četiri ili pet tastera na kojima su iscrtane strelice Već pogađate da ove dirke oslobađaju numeričku tastaturu svih kontrolnih primena — slobodni ste da koristite numerička tastera samo za unošenje brojeva! Ukoliko je vaš računar opremljen još nekim tasterima sa tajanstvenim imenima *Reset CurPad* ili *NoScr* moraćete da bacite pogled u uputstvo za upotrebu vašeg klona ili ako nemate uputstvo da probate!

Pošto smo identifikovali grupe tastera pozabavićemo se dirkama koje imaju specijalne funkcije Ako vam se učini da je rad sa tastaturom strašno komplikovan utešite se nesumnjivom činjenicom da su se vlasnici spektruma mnogo više mučili — vi imate jedan SHIFT a oni ih imaju pet!

SHIFT (ili na PC-ju dvostruka strelica naviše) obezbeđuje kucanje velikih slova — ako držeći taster SHIFT pritisnut pritisnete slovo A na ekranu se pojavljuje veliko A Oba SHIFT-a su u normalnom radu ravnopravna premda AT može posebno da testira svaki od njih

Caps Lock omogućava kucanje velikim slovima — posle pritiska na *Caps Lock* pali se jedna od sijalčica i računar prestaje da detektuje mala slova — pritisak na A će proizvesti veliko A bez obzira na SHIFT Samo se po sebi razume da je taster SHIFT i dalje aktivan — kako biste bez njega otkucali uzvičnik koji je smešten iznad broja 1? Ponovni pritisak na *Caps Lock* vraća računar u normalno stanje, sijalica se gasi Dodajmo da PC u normalnom radu vrlo retko pravi razliku između malih ili velikih slova koristite ona koja vam se više dopadaju

Ctrl obezbeđuje svakom tasteru još jednu funkciju koju pojedini tekst procesori (naročito *WordStar*) rado koriste Za većinu programa je na primer deset funkcijskih tastera nedovoljno pa se ovaj broj učetvorostručuje F1 u uputstvu označava pritisak na F1 S1 pritisak na SHIFT i F1 C1 pritisak na CTRL i F1 a A1 pritisak na Alt i F1

Alt omogućava unošenje ASCII kodova koji nisu pristupačni na tastaturi Ukoliko na primer želite da vidite veliko grčko slovo Omega pritisnite Alt i držeći ovaj taster pritisnut na **numeričkoj** *tastaturi otkucajte brojeve 2 3 i 4 Alt puštate tek pošto otpustite broj 4 Obzirom da je 234 kod velikog Omega na ekranu će se pojaviti baš ovaj znak Taster Alt se koristi i za unošenje mnogih kontrolnih sekvenci*

Backspace je taster na kojem je nacrtana podebljana strelica na levo obično je postavljen u gornji desni ugao tastature baš iznad dirke ENTER Pritisak na *backspace* omogućava brisanje poslednjeg unetog slova — ako otkucate pogrešan znak pritisnite *backspace* i znak će nestati Uzastopnim pritiscima na *backspace* možete da obrišete čitavu komandnu liniju

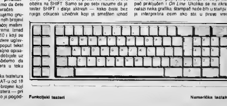
Esc se u komercijalnim programima uglavnom koristi za vraćanje u osnovni meni ili za prekid rada programa Upotreba ovog tastera pri radu sa DOS-om je minimalna — ako primetite da je linija koju ste kucali pogrešna toliko da je nema smisla ispravljati pritisak na *Esc* je mnogo bezbolnija solucija od višestrukog pritiskanja tastera *backspace*

Pomoću tastera *Num Lock* biramo način korišćenja numeričke tastature Kada uključimo računar numerički tasteri imaju funkcije *Insert Delete PgUp* itd Ukoliko želimo da unosimo brojeve pritisnućemo *Num Lock* i početi sa radom Ponovni pritisak na *Num Lock* vraća numeričkim tasterima kontrolne funkcije Stanje *Num Lock-a* prikazuje druga sijalčica

Taster *Num Lock* kada se kombinuje sa CTRL ima još jednu funkciju — zaustavlja skrolovanje teksta Ukoliko neki program prebrzo ispisuje rezultate na ekranu pritisnućemo CTRL i *Num Lock* i ispisivanje će prestati Kada proučimo ispis pritiskamo bilo koji taster i ispisivanje se nastavlja

Taster *Scroll Lock* se koristi isključivo u kombinaciji sa CTRL i tada prekida izvršavanje tekućeg programa Ukoliko se desi da neki program koji ste startovali zaglavi pritisnite CTRl i *Scroll Lock* (ili u daljem tekstu, CTRL Break jer se na tasteru *Scroll Lock* obično nalazi i natpis Break) i nadajte se da ćete videti prompt

Pritisak na SHIFT i *PrtSc* prenosi tekst sa ekrana na štampač pretpostavlja se da je štampač priključen i *On Line* Ukoliko se na ekranu nalazi neka grafika štampač neće biti u stanju da je interpretira osim ako ste u pravo vreme



Funkcijski tasteri

Numerička tastatura

instalirali odgovarajući rezidentni program na primer GRAPHICS

Pritisak na CTRL i *PrtSc* zahteva od računara da sav tekst koji se pojavljuje na ekranu (kako vaše komande tako i odgovori PC-ja) bude prenesen na štampač Na ovaj način imate trag o onome što ste radili što znači da vam je docnije pronalaženje i otklanjanje grešaka mnogo jednostavnije Ispisivanje se prekida ponovnim pritiskom na CTRL i *PrtSc*

Taster *SysReq* na PC-ju ne postoji dok se na AT-u vrlo retko koristi — videli smo da pritisak na ovaj taster izaziva interapt i skreće pažnju operativnom sistemu na zahtev korisnika DOS ignoriše ovaj interapt dok najveći deo postojećeg komercijalnog softvera nije pripremljen da ga registruje Zbog toga pritisak na *SysReq* uglavnom nema nikakvo dejstvo

Kada se kompjuter zaglavi

S vremena na vreme će se događati da kompjuter jednostavno otkaže poslušnost startovali ste neki neispravan ili neispravno instaliran program, čekate a ne dešava se baš ništa! Prvo što treba pokušati jeste pritisak na CTRL i Break (pokušajte i sa CTRL C ako koristite MS DOS a ne PC DOS) što će obično prekinuti program i ispisati prompt Ukoliko više pritisaka na CTRL i Break ne da nikakve rezultate moraćete da resetujete računar *Reset* je relativno neprijatna operacija jer uništava sadržaj kompletne memorije računajući tu i sadržaj takozvanog RAM diska podaci na disketama i hard disku su jasno, bezbedni Jedan od načina za resetovanje je isključivanje i ponovno uključivanje napajanja Daleko je međutim brže pritisnuti tastere Alt CTRL i *Del* (najpre levom rukom pritisnete Alt i CTRL (svejedno kojim redom) a onda držeći ova tastere pritisnute desnom rukom pritisnete i otpustite *Del*) i tako softverski resetovati kompjuter — sve se briše ali bar računar ne testira memoriju i periferijske uređaje

Ponekad vam čak ni tasteri Alt CTRL i *Del* neće pomoći u tom slučaju morate da izvršite hardverski reset Neki klonovi (npr Olivetti M24) imaju *hard reset* taster koji je smešten na samu kutiju računara (nije na tastaturi) dok se na originalnom PC-ju i AT-u hardverski reset svodi na isključivanje i ponovno uključivanje kompjutera Dirka RESET na pojedinim tastaturama predstavlja samo zamenu za kombinaciju Alt-*Del* — pritisak na CTRL i RESET izaziva softverski reset

Korisnici PC-ja treba da izbegavaju resetovanje kompjutera — ovo je operacija previše radikalna i može da rezultira nezatvorenim datotekama i gubitkom dela podataka na disku Računar ćete dakle resetovati samo kada nemate drugog izlaza u normalnom radu treba regularno napustiti korišćeni program (izaberete na primer, opciju *Return to DOS* iz menija tekst procesora) i kada se pojavi prompt ugasiti kompjuter

Editovanje komandne linije

Često će vam se dešavati da otkucate neku liniju pritisnete ENTER i ugledate poruku o greški Greška je verovatno bila sasvim sitna izostavili ste neko slovo ili specijalni znak Mogli biste jasno da prekucate čitavu komandnu liniju uz malo više pažnje, tako rade mnogi početnici Ako je linija bila iole duža ovo je prekucavanje neprijatno pogotovu ako znate da nije nužno DOS je opremljen editorskim komandama koja omogućavaju ispravljanje greške Predlažemo vam da odmah investirate malo napora u savladavanje editora ovo će vam znanje u budućnosti uštedeti silno kucanje

Komanda koju ste otkucali se bez obzira na to da li je ispravna ili ne smešta u specijalni bafer iz koga možete da je čitate slovo po slovo Najjednostavnije je pritisnuti F3 čime se cela linija čita iz bafera — na ekranu se pojavljuje



naredba koju ste prethodno otkucali i računar očekuje da pritisnete ENTER da bi je izvršio Taster F3 ne morate da pritiskate samo ako je linija bila pogrešna ponekad treba da ponovite prethodnu (sasvim korektnu) naredbu

Kada se, po pritisku na F3 linija pojavi na ekranu možete da je ispravljate da dodajete slova ili da, pritiskom na *backspace* brišete njen kraj Ukoliko se greška u kucanju nalazila na kraju linije na ovaj način brzo ćete je ispraviti Šta međutim da se radi ako je greška bila na početku?

Došlo je vreme da upoznamo funkciju tastera F1 koji iz bafera prenosi samo jedan znak Otkucali smo na primer FOTMAT A i računar je ispisao *Bad command or file name* — naredbe FOTMAT ne postoji Pritisnemo F1 i na ekranu se pojavi slovo F Još jedan pritisak na F1 daje slovo O, sledeći bi pritisak proizveo slovo T koje je pogrešno Zato pritiskamo slovo R a zatim F3 — na ekranu piše FORMAT A i računar čeka da pritisnemo ENTER Naučili smo dakle, da ispravimo jedno ili nekoliko pogrešno otkucanih slova

Šta da radimo kada umesto FORMAT A otkucamo FORRMAT A — jedno R je višak Tri puta pritisnemo F1 i na ekranu vidimo tekst FOR Zatim pritisnemo *Del* i tako obrišemo slovo R, ostalo je još da pritisnemo F3 i tako završimo liniju

Ni umetanje teksta nije mnogo komplikovanije Otkucali smo na primer FORM A Četiri pritiska na F1 proizvode tekst FORM Zatim pritiskamo *Insert* i kucamo slova A i T a zatim sa F3, proizvodimo ostatak linije

Tasteri f1 F3 *Insert* i *Del* su verujte nam na reč sasvim dovoljni za komforno editovanje linija Slika 3 međutim pokazuje da su nas autori DOS-a opremili i nekim luksuznim opcijama Umesto da kao u prethodnom primeru četiri puta kucamo F1 mogli smo da pritisnemo f2 a zatim blanko i računar bi iz bafera pročitao sva slova do blanka ako je linija duža a greška u njenoj sredini na ovaj način dosta štedimo na kucanju (ne zaboravite da F1 kao i svaki drugi taster ima *auto repeat* — ako ga držite pritisnut računar brzo prenosi sukcesivne znakove)

Vredi još pomenuti taster F5 koji tekuću liniju prenosi u bafer ali ne izaziva njeno izvršavanje Ukoliko ste dakle u toku kucanja neke duge komandne linije primetili grešku na njenom početku pritisnite F5 i zatim editujte liniju na standardan način tako ste uštedeli DOS-u trud da pronađe i prijavi grešku Tastere F4 i F6 ćete vrlo retko koristiti

Iako koristan PC-jev komandni editor nije baš naročit — omogućava samo ispravljanje prethodne komande pri čemu je rad sa baferom prilično tajanstven jer korisnik ne vidi na koji znak u njemu skriveni kurzor pokazuje Zato je korisno nabaviti neki alternativni komandni editor i od samog se početka navikavati na njega Jedan od takvih se zove CED

Komande DOS-a

Sledeće stranice posvećujemo komandama MS DOS-a koja ćete svakodnevno koristiti

slika 3

ESC	Briše tekuću liniju ne menjajući sadržaj bafera
INS	Omogućava umetanje jednog ili više znakova
DEL	Briše jedan znak iz bafera
F1	Kopira jedan znak iz bafera u tekuću liniju
F2	Kopira znake iz bafera zaključno sa otkucanim
F3	Kopira ostatak linije iz bafera u tekuću liniju
F4	Briše sve znake iz bafera do otkucanog karaktera
F5	Poništava kompletnu liniju i prenosi je u bafer
F6	Proizvodi marker kraja datoteke (kao CTRL Z)

BACKUP — čuvanje sadržaja datoteka Naredba *BACKUP* omogućava prenošenje sadržaja diskete na disketu diskete na hard disk, hard diska na diskete ili hard diska na hard disk Osnovni cilj ovakvog prenošenja podataka je sigurnost — ukoliko se hard disk ili disketa oštete podaci su sačuvani Operacija *BACKUP* treba izvršavati onoliko često koliko to vreme dopušta — ukoliko se svakodnevno kucaju opširni tekstovi ili velika količina podataka neophodan je dnevni *backup*, ukoliko, sa druge strane PC koristite kod kuće ovu operaciju ćete obavljati mnogo ređe na primer nedeljno ili mesečno

Najjednostavniji način da prenesete sadržaj hard diska na diskete je BACKUP C A /S Pre kucanja ove komande pripremite dovoljno praznih disketa za kopiranje popunjenog hard diska od 20 megabajta potrebno je nekih 57 disketa po 360 K (vlasnici AT-a će naravno, koristiti 16 disketa od po 1 2 megabajta) Korisnici koji smatraju da je ovakvo kopiranje zamorno često prenose takstove programe i podatke koje su sami pripremili na diskete običnom komandom COPY a komercijalne programe čuvaju u izvornom obliku na disketama pa im je BACKUP nepotreban Nevolje se javlja kada neka od datoteka sa podacima prerasta 360 kilobajta — njeno prenošenje sa COPY nije moguće U tom slučaju treba koristiti BACKUP za jednu datoteku (bez džoker znakova)

CHDIR (CD) — promena radnog direktorijuma Naredbu detaljno opisujemo u sledećem poglavlju Ukratko iza CHDIR navodimo kompletno ime direktorijuma koje možemo da zadajemo u odnosu na radni direktorijum ili na *root.* CHDIR uvek možemo da zamenimo sa CD — tako štedimo kucanje

CHKDSK — provera ispravnosti diska S vremena na vreme treba kontrolisati integritet podataka na disku — komanda CHKDSK, osim proste provere omogućava i ispravljanje pronađenih grešaka Treba razumeti da se ovom komandom ne verifikuje disk — proverava se samo katalog i tablica alociranog prostora. Ukoliko računar pronađe prostor alociran za neke datoteke koje se ne pominju u direktorijumima, biće omogućeno uključivanje ovih programa u opštu shemu Da bi se ovakva izmene stvarno izvršile treba otkucati CHKDSK /F

Sama komanda CHKDSK prikazuje sledeće informacije ime diska i datum zadnjeg formatiranja raspoloživi prostor na disku prostor koji zauzimaju skrivene datoteke prostor koji zauzimaju potkatalozi prostor koji zauzimaju datoteke prostor propao zbog neispravnosti diska raspoloživ prostor na disku ukupan kapacitet RAM-a i kapacitet slobodnog RAM-a

COPY — kopiranje datoteka Naredbu COPY ćete koristiti vrlo često — uz njenu pomoć se datoteke prenose sa diska na disk i iz direktorijuma u direktorijum Sintaksa je COPY *izvor odredište* a za racionalno korišćenje ova komanda treba dobro upoznati džoker znake



DATE — rad sa kalenderom. Originalni IBM PC i IBM PC XT nisu opremljani časovnikom realnog vremena ali DOS na principu interapta neprekidno računa tačno vreme koja obuhvata dan datum mesec godinu sata minute i sekunde. Vreme se, jasno, gubi kada se računar isključi što znači da po svakom uključivanju treba da doterate časovnik i kalendar. IBM PC AT (i po ugledu na njega mnogi PC kompatibilci) je opremljen baterijski napajanim časovnikom realnog vremena što znači da sat i kalendar rade i dok je kompjuter isključen. Doterivanje je potrebno samo s vremena na vreme i to naredbom DATE. Da biste na primer saopštili računaru da je danas 17 decembar 1987 otkucaćete DATE 17—12—1987 ili ako koristite američki format prikazivanja vremena DATE 12—17—1987. Kucajući samo DATE možete da saznete tekući datum

DEL — brisanje datoteka. Komanda DEL je potpuno identična komandi ERASE i služi za brisanje datoteka

DIR — podaci o datotekama u tekućem katalogu. Komandom DIR prikazujemo sledeće podatke: ime diska (ako postoji), kompletno ime direktorijuma podatke o datotekama (ime, dužina i datum kreiranja odnosno poslednje modifikacije), ukupan broj prikazanih datoteka i preostali slobodan prostor na disku. Ako iza DIR dodamo /P ispisivanje će prestati kada se ekran popuni — računar čeka da pritisnemo neki taster da bi nastavio listanje. Sa DIR /W zahtevamo da se prikaže samo minimum podataka o svakoj datoteci — tako na ekran može da stane i sadržaj veoma popunjenog direktorijuma

DISKCOPY — kopiranje diskete. Primenom komande DISKCOPY prepisujemo kompletan sadržaj jedne diskete na drugu. Podrazumeva se da se radi o standardnoj DOS disketi, zaštićene programe morate da kopirate na drugi način. Sintaksa je DISKCOPY izvor odredište

Ako odredišna disketa nije formatirana, PC će je automatski formatirati. Za ovu je operaciju, jasno, potrebno neko vreme. Sa ili bez formatiranja kopiranjem se potpuno i nepovratno uništava raniji sadržaj odredišne diskete — budite vrlo oprezni!

ERASE — brisanje datoteka. Komandom ERASE brišemo nepotrebnu datoteku ili (uz džoker znake) grupu datoteka. Pre nego što iskoristite komandu ERASE da biste, uz pomoć džoker znakova, obrisali grupu datoteka nije loše otkucati DIR kako biste videli koje će sve datoteke biti obrisane. Ukoliko nehotice obrišete datoteku koja vam je potrebna pokušajte da je povratite pomoću programa Norton Utilities. Važno je da od momenta kada ste primetili grešku ništa ne upisujete na disketu ili disk — tako čuvate sadržaj obrisane datoteke

FORMAT — formatiranje diska. Pre upotrebe svaki disk ili disketu treba formatirati. Hard disk C: se *uvek* formatira sa *format C:/S/V* dok je formatiranje disketa mnogo raznovrsnije. Razlog za ovu raznovrsnost treba tražiti u istoriji DOS-a: kapacitet disketa se više puta menjao što znači da današnja FORMAT naredba mora da priprema diskete čitljive na raznoraznim verzijama operativnog sistema. Najjednostavnije je da otkucate FORMAT A: i kada računar to zahteva umetnete disketu u drajv A. Vlasnici AT-a i drajva od 1.2 megabajta sa FORMAT A: /4 formatiraju disketa koja će (eventualno) biti čitljive na običnom PC-ju

Ukoliko formatirate disketu sa koje će se računar docnije „podizati", tj. koja će sadržati kopiju MS DOS-a, otkucajte FORMAT A: /S. Sa FORMAT A: /V računar vam omogućava da disketi koju formatirate date ime koje će vas podsećati na njen sadržaj. Opcije /4 /S i /V mogu slobodno da se kombinuju

Formatiranjem diskete se nepovratno uni-



štavaju sve informacije na njoj — ne postoji program koji može da povrati izgubljene podatke! Budite, prema tome, veoma obazrivi

MKDIR — kreiranje kataloga. MKDIR je naredba koja omogućava formiranje novog kataloga na način koji ćemo opisati u sledećem poglavlju. Iza MKDIR ukratko navodimo kompletno ime kataloga koje možemo da zadajemo u odnosu na radni direktorijum ili na *root*. MKDIR uvek možemo da zamenimo sa MD — tako štedimo kucanje

MODE — podešavanje raznih parametara. Komanda MODE se koristi za mnogo suštinski različitih stvari pa ćemo ovde pomenuti samo par njenih oblika

Sa MODE LPTn: CPL, LPI saopštavamo računaru da štampač vezan na n-ti paralelni port (npr. LPT1) ispisuje *cpl* znakova u redu (npr. 80) i *lpi* linija po inču (npr. 6). Pretpostavlja se da je štampač Epson ili IBM kompatibilan

Sa MODE LPTn=COMm (npr. LPT1=COM1) nalažemo računaru da sav tekst koji ubuduće bude poslat paralelnom štampaču prosledi serijskom — možda tekst procesor koji koristimo ne podržava rad sa više printera!

Sa MODE COMM: BAUD PARITY, DATABITS, STOPBITS konfigurišemo serijski port n (npr. COM1): *baud* predstavlja brzinu prenosa (110 150 300, 1200 2400 4800 ili 9600 bauda) *parity* (N<none>, O<odd> ili E<even>) nalaže računaru da vrši ili ne vrši kontrolu parnosti, *databits* označava koliko svaki preneti bajt ima korisnih bitova (7 ili 8) a *stopbits* označava broj stop bitova na kraju svakog bajta (1 ili 2)

PARK — parkiranje glave hard diska. Hard disk se u toku rada računara stalno okreće a glave mu pristupa samo kada su podaci potrebni. Isključivanjem kompjutera jasno prestaje i rotacija diska i samim tim postoji mogućnost da glava padne na magnetni medij i ošteti ga. Zato neki proizvođači predlažu da se glava pre gašenja kompjutera premesti iznad nekog nekorišćenog dela diska, za to se koristi program PARK

PATH — direktorijumi u kojima treba tražiti komandu. Kada otkucate neku naredbu, komandni interpreter najpre pokušava da je pronađe među internim komandama. Ukoliko ne uspe pokušava da izvrši istoimenu COM, EXE ili BAT datoteku u tekućem direktorijumu. Ako ni jedna od ovih datoteka ne postoji, DOS bi mogao da prijavi grešku. Omogućeno je međutim definisanje direktorijuma koje će DOS dodatno pretraživati u potrazi za specificiranom COM, EXE ili BAT datotekom. Ovi se direktorijumi specificiraju komandom PATH. U sledećem poglavlju videćemo zašto su direktorijumi potrebni i kako da ih kreiramo. Za sada ćemo zapamtiti da je sintaksa komanda PATH *dir1; dir2, dir3;...*

RENAME (REN) — promena imena datoteke. Naredba RENAME ili njena ekvivalentna skraćenica REN koriste se za promenu imena grupe datoteka. Za efikasnu upotrebu treba se dobro upoznati za džoker znacima. Sintaksa je RENAME *disk direktorijum ime1 ime2* — datoteka *ime 1* se preimenuje u *ime 2*

Naredbom RENAME možete samo da menjate imena datoteka u okviru jednog direktorijuma — ne možete da prenosite fajl iz jednog direktorijuma u drugi a još manje sa diska na disk

RESTORE — vraćanje datoteka sačuvanih sa BACKUP. Upoznali smo naredbu BACKUP koja omogućava prenošenje sadržaja datoteka na diskove ili disketa koje ćemo koristiti ako osnovni podaci na neki način stradaju. Naredba RESTORE vraća ovakve datoteke na hard disk ili disketu. Dok ćemo naredbu BACKUP izvršavati onoliko često koliko to vreme dopušta (možda i svakodnevno) nadaćemo se da nam RESTORE nikada neće zatrebati! Ako se želja ne ispuni, koristićemo verovatno RESTORE A: C: /S

RMDIR (RD) — uklanjanje direktorijuma. Komandom RMDIR ili njenom ekvivalentnom skraćenicom RD uklanjamo nepotreban direktorijum. Direktorijum mora da bude potpuno prazan da bismo ga obrisali! Programi kao što je dBASE ili imaju ružu osobinu da u radnom direktorijumu kreiraju skrivene datoteke koje sa DEL ne možete da obrišete. Kada se radi o dBASE-u ove datoteka brišete program UNISTALL. Ako skrivene datoteke nastane na neki drugi način koristite



Norton, setujte se u *roditeljski* direktorijum i „obradite" odgovarajući red u DBASE <DIR>

TIME — prikazivanje vremena i doterivanje časovnika. Naredba TIME je slična sa DATE s tim što umesto doterivanja datuma obezbeđuje doterivanje vremena. Ako pogledom na (pravi) časovnik ustanovite da je upravo 18 časova, 15 minuta i 10 sekundi, otkucaćete TIME 18,15 10.00. U zavisnosti od sadržaja datoteke CONFIG.SYS separator između sekundi i stotinki može da bude i zarez

TYPE — ispisivanje sadržaja datoteka. Iako o naredbi TYPE ne može mnogo da se piše vrlo ćete je često koristiti: uz njenu pomoć se sadržaj neke ASCII datoteke prenosi na ekran. Sintaksa je TYPE *ime_datoteke*. Ako datoteke sadrži kontrolne kodove ponašanje je nepredvidljivo — računar može i da se blokira. U ovom slučaju pritisnite CTRL Break ili, u gorem slučaju, resetujte sistem

KREATIVNOST U OBR

MREŽE Z

Danas se često govori o revolucionarnim promenama u proizvodnji, proizvodnim sredstvima i metodama u svim oblastima rada i interesovanja.

Zbog ovog trenda škole su dobile dva osnovna zadatka. Prvi, neposredno proističe iz njihovog vaspitnog karaktera, da se učenici pripreme i upoznaju sa novim elektronskim uređajima sa kojima će doći u kontakt. Drugi proističe iz činjenice da i škole predstavljaju jednu specifičnu radnu sredinu, koju je neophodno opremiti savremenim uređajima u cilju usvajanja potrebnih metoda za njihovo korišćenje.

Mreža za učionice je aplikativno područje, bazirano na lokalnoj mreži (LAN), sa povezivanje više razrada, povezivanjem serije personalnih računara. Sistem je organizovan na hijerarhijskoj bazi i u stanju je da rukovodi sa povezanim radnim mestima i zajedničkim resursima.

Bez obzira na to što mreža predstavlja savremeniju metodu u poređenju sa tradicionalnim metodama učenja, korišćenje ovog rešenja ne zbunjuje učenike i nastavnika. Ako uzmemo u obzir ovu bitnu činjenicu, da se radi o kombinovanju nove i stare metode učenja, onda je razumljivo da nova procedura pruža brže i kreativnije učenje.

Tip strukture koji vam predstavljamo ima značajne ekonomske i funkcionalne prednosti jer u okviru svih lokalnih mreža omogućuje:

— sa svakom radnom mestu istovremeni pristup svim informacijama koje sadrži sistem;

— međusobno povezivanje različito povezanih personalnih računara;

— koristi softver koji je prilagođen specijalnim oblastima i može se aktivirati pomoću tipki obične tastature.

Zahvaljujući visokoj tehnologiji i činjenici da je sistem posebno rađen za vaspitni sektor, mreža obezbeđuje operativne odlike koje se razlikuju od standarda uobičajene mreže, kao što je mogućnost prikazivanja, zatim trenutno prikazivanja rada svakog pojedinačnog radnog mesta, konverzaciju između nastavnika i pojedinih učenika, intervenciju nastavnika po potrebi, kao i korišćenje tehnike „elektronske table".

Struktura

Rešenje za mrežu dali su stručnjaci Novkabela i Olivetti-ja. Sistem sadrži neodređeni broj personalnih računara Olivetti, koji koriste MS-DOS operativni sistem.

Sistem sadrži tri osnovne dele

— Radno mesto učenika sadrži jedan personalni računar Olivetti koji može da koristi poseban softver učenje i resurse mreže

— Radno mesto nastavnika sadrži jedan personalni računar Olivetti. Nastavnik može postupno da prikaže lekciju, kao i da porukama ili sugestijama interveniše kod učenika, ako se se to ukaže potreba

— Server, ili uslužna jedinica, sadrži jedan personalni računar Olivetti u konfiguraciji „hard diska", uz poseban softver za učenje i upravljanje periferním jedinicama, a to su

— magnetna memorija za datoteke učenje

— štampač

— ploter

U mreži može da se rukovodi sa promenljivim brojem povezanih personalnih računara (najviše 25), ali najpovoljnije strukture za efikasno rukovođenje po merilima učenja je 15 personalnih računara. Jednostavnost operativnih naredbi, kao i činjenica da se može koristiti bilo koji MS-DOS aplikativni program, obezbeđuje upotrebu ove mreže u bilo kojem obrazovnom području — na fakultetu, u srednjoj školi, u osnovnoj školi — ili za samostalno vežbanje kod kuće

Operativna odlike

Predviđena struktura predstavlja prelomnu tačku u metodama učenja, a bazira se na sledećim operativnim odlikama.

— Na jednostavnosti u primeni, pošto se sve operacije kontrolišu pomoću jednostavnih naredbi i menija koji ih objašnjavaju.

— Na prikazivanju rada učenika. Nastavnik može da bira ili da pozove sve što se nalazi na displeju svakog pojedinog učenika, u cilju da proveri tok lekcije, da izvrši korekcije, da memoriše tačne zadatke u datoteke i da vrati iste učeniku

— Na dijalogu nastavnik—učenik. Nastavnik može privremeno da obustavi aplikativno područja učenika, kao i da pošalje sugestije i pitanja na koje učenik može odmah da odgovori pre završetka rada

— Na porukama celom razredu. Nastavnik može da pošalje poruke objašnjenje celom razredu

— Na električnoj tabli, sistem omogućuje nastavniku da prikaže na ekranima svih radnih mesta primer koji je sam pripremio, kao i da prekine sa prikazivanjem primera u svakom trenutku i da zahteva od nekog izabranog učenika da sam nastavi primer, dok drugi učenici prate daljni tok lekcije.

— Na punjenju sistema programima servera, svako radno mesto može lokalno da koristi bilo koji program koji se nalazi u uslužnoj jedinici, bez potrebe preplaćivanje istog na disketu

— Na pristupu zajedničkim resursima, sa svakog radnog mesta moguć je neposredan pristup bilo kojoj jedinici povezanoj sa serverom (štampačima, ploterima, masovnim memorijama, itd).

— Na memorisanju datoteke u uslužnoj jedinici, ako učenik ili nastavnik želi da sačuva uređen red u cilju posebnog rezimea, ili za potrebe statističke dokumentacija, on može da koristi velik kapacitet memorije uslužne jedinice kao sistema opšteg memorisanje

— Na povezivanju sa ostalim mrežama. Mreža je bazirana na MS-NET standardu koji koriste sve lokalne i udaljene mreže, i zbog toga se može povezati sa ostalim mrežama i time imati pristup specijalnim spoljnim bankama podataka ili telesoftverskim uslugama.

Čak i ovakav kratak rezime može da pomogne u sagledavanju budućeg razvoja učenja. To je mnogo bliže nego što se može pretpostavljati, jer se posebno područni i aplikativni softveri već nalaze na raspolaganju. Štaviše, oni ne zahtevaju posebno znanje za njihovo primenjivanje, kao ni posebnu upućenost u metode kompjuterskog programiranje.

INFORMACIONI SISTEM ŠKOLA — ĐAK

Informacioni sistem ĐAK pruža automatizovanu organizacionu podršku radu škola u aktivnostima koje okružuju pedagoško-obrazovni proces. Realizuje se na personalnim računarima ET—188A i otvoren je za dalji razvoj. Sastoji se od sledećih podsistema

ADMINISTRACIJA

Sprovođenja konkursa za prijem učenika — **PRIJEM**

Evidencija i statistika učenika — **SEKRETAR**

KNJIGOVODSTVO

Finansijsko knjigovodstvo sa kupcima i dobavljačima —

NOVAC

Obračun ličnog dohotka — **PLATA**

Evidencija osnovnih sredstava — **KLUPA**

Evidencija sitnog inventara — **ALAT**

RADIONICA

Lansiranje radnih naloga — **URADI**

Materijalno knjigovodstvo — **UGRADI**

Fakturisanje usluga — **NAPLATI**

BIBLIOTEKA

Evidencija školske biblioteke — **KNJIGA**

Osnovna namena aplikacija informacionog sistema ĐAK

PRIJEM — evidentiranje i rangiranje učenika prema uslovima konkursa uvažavajući alternativne želje učenika

SEKRETAR — evidentiranje brojnog stanja učenika u odeljenjima po raznim obeležjima uspeh socijalno poreklo nacionalna struktura mesto stanovanja i drugo. Štampanje statističkih izveštaja po navedenim obeležjima.

NOVAC — zaključni list glavna knjiga sintetička i analitička evidencija kontni plan registar poslovnih partnera izvod otvorenih stavki

PLATA — obračun ličnog dohotka sa bruto na neto evidencija kredita uplata bankama automatski obračun doprinosa i ostalih plaćanja štampa isplatnih i obračunskih lista virmana i analiza

KLUPA — evidentiranje i održavanje spiska osnovnih sredstava obračun amortizacije i revalorizacije analitički pregled i drugi izveštaji

ALAT — zbirna evidencija sitnog inventara periodične amortizacije i revalorizacija analitički pregled i drugi izveštaji

URADI — evidencija prijema u radionicu praćenje knjige radnih naloga lansiranje radnih lista

NAPLATI — evidentiranje izvršenih operacija i automatsko povezivanje sa ugrađenim materijalima obračun usluga sa izradom računa praćenje naplate

Oglašavanje
RO Novosadska fabrika kabela
OOUR Elektronski računari

Put Novosadskog partizanskog odreda 4
Telefon 021-338-199/2211
337-256 direktni

AZOVANJU

ZA UČIONICE

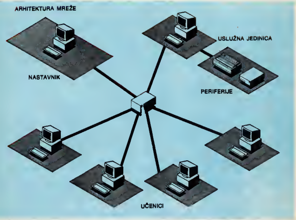
NOVKABEL

NFK

RAČUNARI NOVKABEL

PREGLED KARAKTERISTIKA

	FRS 51XX	FRS 71XX	FRS 81XX
CPU	8088-2 4,77/8 MHz	8088-2 4 77/8 MHz	80286-2 6/10 MHz
RAM	640 KB	640 KB	1 MB
Jedinice disketa	2×360 KB	1×360 KB opcija 1×360 KB	1×1 2 MB opcija 1×1,2 MB
Jedinice diska	opcija 2×20 MB	1×20 MB opcija 1×20 MB	1×40 MB opcija 1×40 MB



RZ KOMERCIJALNI POSLOVI
Poslovnica prodaje
Tel. 021/337-255
21000 Novi Sad
Put Novosadskog partizanskog
odreda br. 4

Otvoreni ili vođeni časovi, koji su rezultat istraživanja fakultetskih stručnjaka u posebnim oblastima, i paketi autoprogramiranja koji ugrađuju zahteve nastavnika u program, već su prevazišli eksperimentalnu fazu i nalaze se na raspolaganju nastavnika i učenika u cilju usklađivanja pristupa učenja sa evolucijom vremena.

Datoteke na PC-ju

Videli smo da se rad sa OOS-om svodi na kucanje komendi i startovanja programa. Programi koje startujemo smešteni su u takozvane datoteke (u stranoj literaturi *files* fajlovi); na vašoj OOS disketi dobili ste priličan broj datoteka sa programima, a uskoro ćete i sami početi da pravite datoteke sa tekstom i podacima.

Pokušajmo da kreiramo jednu datoteku. Otkucaćemo COPY CON PROBA BAT i pritisnuti ENTER. Zatim otkucamo *ECHO Ovo je probni ispis*, pritisnemo ENTER, CTRL Z i još jednom ENTER. Na ekranu je ponovo uobičajeni prompt. Otkucamo OIR i, među ostalim datotekama, otkrijemo našu PROBA BAT. Šte je u datoteci? Otkucamo TYPE PROBA BAT, pritisnemo ENTER i računar ispisuje sadržaj datoteke, naredbu 'ECHO Ovo je probni ispis'. Ova datoteka je zapravo mali program koji možemo i da izvršimo otkucaćemo samo PROBA i na ekranu se pojavljuje tekst 'Ovo je probni ispis'. Datoteka opstaje na disku sve dok je ne obrišemo sa ERASE PROBA BAT.

Pošto smo kreirali jednu datoteku, imamo barem grubu predstavu o tome šta reč *datoteka* znači: radi se o skupu međusobno povezanih podataka koje smo objedinili i smestili u zajedničku „fasciklu" (*file*=fascikla, naša reč datoteka potiče od *teka* (=sveska) *sa podacima* (=data)). Naša datoteka se zove PROBA, tip joj je BAT i sadrži (jednu) naredbu upravljačkog jezika.

Imena datoteka

Ime datoteke sastoji se iz dva dela: naziva i tipa. Naziv je bilo koja reč od najviše osam slova, brojki i specijalnih simbola (dopušteni su samo neki specijalni simboli a predlažemo vam da se i njih odreknete) dok se tip sastoji od jednog, dva ili tri slova. Između naziva i tipa uvek stavljamo tačku. Zašto je uz naziv datoteke potreban i njen tip? Kada počnete da radite sa tekst procesorom, napisaćete neko pismo i inventivno ga nazvati PISMO1 TXT. Čim prvi put počnete da ispravljate pismo, računar će datoteku PISMO1 TXT preimenovati u PISMO1 BAK (BAK dolazi od BAcKup, rezervna kopija) što znači da ćete, ako ustanovite da su izmene loše ispale, uvek moći da se vratite na originalnu verziju. Obzirom da datoteke PISMO1 TXT i PISMO1 BAK praktično predstavljaju dve faze istog posla, logično je da se obe zovu PISMO1, ali da im je tip (ili kako se ponekad kaže ekstenzija) iste. Autori PC OOS-a predvideli su standardne ekstenzije za neke tipove datoteka (COM i EXE su izvršni programi, BAT nizovi upravljačkih naredbi i tome slično), ali u stvari bilo koja tri slova mogu da predstavljaju tip. Nema, dakle, nikakve prepreke da izmišljate tipove u skladu sa svojim potrebama ili svojom imaginacijom. Ipak, standardni tipovi COM, EXE i BAT imaju tačno određeno značenje, pa vam ne savetujemo da neki tekst nazovete PISMO COM — tako dovodite u zabunu kako sebe tako i druge. Prilično je, dalje, uobičajeno da tip ima tačno tri slova, premda nema nikakve prepreke da bude i kraći — gotovo uz svaki komercijalni program dobijate datoteku READ ME koju obavezno treba da pročitate (otkucajte TYPE READ ME ili, ako imate štampač, COPY READ ME LPT1). Datoteka ne mora uopšte da ima tip, ali vam ne preporučujemo da kreirate previše ovakvih fajlova, samo ćete se nepotrebno zbunjivati!

Pošto smo naučili šta su tipovi, pogledajmo sliku 4 na kojoj je dato nekoliko propisnih i nekoliko nepropisnih imena datoteka. Proverite svoje znanje tako što ćete, ne čitajući objašnjenja, zaključiti zašto je ime neke datoteke pogrešno.

Džoker znaci

Najlakši način da pristupite nekoj datoteci je, naravno, da u odgovarajućem kontekstu navedete njeno ime. Da biste, na primer, na ekran preneli sadržaj datoteke sa tekstom koja se zove PROBA, otkucaćete TYPE PROBA TXT (pokušajte da ovako ispišete sadržaj neke COM ili EXE datoteke i ekran će se napuniti „dubretom". Pritisnite CTRL i Break da se ispisivanje prekine). Često je, međutim, potrebno izvršiti neku operaciju nad grupom datoteka — želimo, na primer, da izbrišemo već pomenute fajlove PISMO1 TXT i PISMO1 BAK. Mogli bismo, naravno, da otkucamo ERASE PISMO1 TXT a zatim ERASE PISMO1 BAK — volite li da kucate? Ako ne volite, otkucajte ERASE PISMO1 • — zvezdica je ovde takozvani džoker znak koji zamenjuje bilo koji nastavak („džoker znak" je, inače, zgodan prevod engleskog izraza *wild character*).

MS OOS podržava dva džoker znaka: znak pitanja zamenjuje bilo koji znak dok zvezdica zamenjuje bilo koju *grupu* znakova. U tom smislu bi ERASE PISMO? T? obrisalo datoteke PISMO T (i ništa je znak!), PISMO1 TX, PISMO2 TE i slične dok bi ERASE PISMO? T•, osim ovih, obrisalo i PISMO1 TXT, PISMO2 TTT, PISMOK TOP i slične. U praksi se zvezdica daleko češće koristi — svakodnevno ćemo kucati nešto poput OIR • TXT i tako na ekranu ispisivati imena svih datoteka tipa TXT koje su trenutno upisane na disk. OIR • • bi, pogađajte, prikazalo podatke o svim datotekama na disku ali ovakva upotreba džoker znakova nije neophodna — dovoljno je otkucati DIR.

slika 4

Korektna	Nekorektna	
PROBA FOR	PROBA FORT	tip ima više od tri slova
P123 C	P123/C	neispravan separator
P123	P123	neispravan tip
123 EXE	*123 EXE	nedozvoljen simbol u imenu
PROGRAMI LIB	POTPROGRAMI LIB	više od 8 slova u imenu
NOESCC BAT	NOESC/ BAT	nedozvoljen simbol
PROG−1 FOR	PROG_1 FOR	blanko simbol u imenu
TXT TXT	AUX	rezervisano ime uređaja

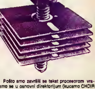
Upotreba džoker znakova u MS DOS-u nije, na žalost, rešena baš savršeno: jedno ime može da sadrži samo jednu zvezdicu (ekstenzija može da sadrži još jednu) koja uz sve to ne sme da se nađe na prvom mestu. Možemo, dakle, da napišemo OIR PR• • ali ne OIR •PR • ili DIR PR•A• OAT.

Hard disk i direktorijumi

Pošto smo razumeli zašto je datotekama uz naziv, potreban i tip, reći ćemo da ni taj tip nije dovoljan — potrebni su i direktorijumi. Korisnici PC-ja koji se (još) nisu opremili hard diskom obično preskaču sve što se odnosi na direktorijume, upoznali smo i takve koji neće da uzmu disketu sa programom koji se prostire u nekoliko direktorijuma! Vlasnici hard diska, s druge strane, **moraju** da znaju šta su to direktorijumi i **moraju** da nauče da ih racionalno koriste. Zašto moraju? Zato što se deset, dvadeset ili četrdeset megabajta podataka jednostavno ne može staviti u jedan katalog!

Dobili ste, na primer, *WordPerfect* na četiri diskete i želite da ga prenesete na hard disk. Ništa lakše: četiri puta ćete otkucati COPY A• •C. Otkucate OIR i vidite datoteke od kojih se *WordPerfect* sastoji. Ima ih dobrih tridesetak a jedna od njih je čuvena READ ME. Sutra dobijete *Lotus 123* — nove tri-četiri diskete. Kucate COPY A• • C:, zatim DIR i vidite mešavinu datoteka koje pripadaju *Lotusu 123* i datoteke koje pripadaju *WordPerfect*-u. Da bi stvar bila još lepša, i *Lotus* ima svoju datoteku READ ME koja je prebrisala istoimenu poruku *WordPerfect*-a. Baš lepo. A zamislite da svakog sledećeg dana dobijete po neki novi program? Direktorijum će se protegnuti preko deset ekrana, neće se znati koja datoteka pripada kom programu, neki će delovi biti prebrisani... Tako se jednostavno ne može!

Setimo se prvog dana kada je u direktorijumu bio samo *WordPerfect* — sve je moglo lako da se nađe i pročita. Kada bi *WordPerfect* imao svoj direktorijum, ne bi nastajala nikakva zabuna. Ništa lakše — otkucaćemo MKOIR WPERF (MKOIR dolazi od Make OIRectory, napravi direktorijum), zatim CHOIR WPERF (CHDIR=CHange OIRectory) i najzad OIR — direktorijum je potpuno prazan ako se izuzmu dve tajanstvene datoteke koje se zovu '.' i '..' (ove datoteke jednostavno zanemarite — nalaze se u svakom potkatalogu i na neki ga način povezuju sa ostalim katalozima na disku). Ako sada otkucamo COPY A• • C: *WordPerfect* će biti prepisan u direktorijum koji je otvoren ekskluzivno za njega — neće se mešati ni sa jednim drugim programom!



Pošto smo završili sa tekst procesorom, vraćamo se u osnovni direktorijum (kucamo CHDIR \) kreiramo novi direktorijum za *Lotus* (MKDIR LOTUS), prelazimo u njega (CHDIR LOTUS) i kopiramo programe (COPY A• • C:). Ponovo se vraćamo u osnovni direktorijum (CHDIR \), kreiramo potkatalog za program dBASE (MKOIR DBASE).

Do sada smo znali da ime datoteke ima dve komponente: naziv i tip. Kada malo bolje razmislimo, tu je i treća komponenta: oznaka diska na koji je datoteka snimljena. Neki tekst bi, dakle, mogao da se zove C GLAVA1 TXT, C je oznaka diska, GLAVA1 je naziv a TXT tip. Sada ćemo naučiti da je jedna od komponenti imena i oznaka direktorijuma u kome se datoteka nalazi,

ova oznaka se piše između imena diska i imena datoteke. Jedna reč utehe: ovim smo iscrpli sve komponente imena datoteke. Da se ne biste baš previše utešili dodaćemo da ovu tvrdnju treba da zaboravite čim povežete PC-je u lokalnu mrežu!

Direktorijum se obično specificira u odnosu na osnovni, takozvani *root* katalog. Šta je to *root* katalog? Ako kataloge zamislite kao stablo, *root* je koren stabla — kada uključite računar i otkucate DIR, na ekranu će se pojaviti sadržaj osnovnog kataloga! U bilo kom trenutku možete da pređete u osnovni katalog tako što ćete otkucati CHDIR#

Osnovni katalog može da sadrži datoteke i druge kataloge. Ti drugi katalozi mogu da sadrže datoteke i treće kataloge i tako do prilično velike dubine. Zbog brzine rada je, međutim, preporučljivo da ova dubina ne pređe 3: imate osnovni katalog, njegove potkataloge i potkataloge tih potkataloga!



Koncipiranje stabla kataloga je odgovorna operacija koja će imati mnogo uticaja na racionalan rad vašeg kompjutera. Pre svega, osnovni katalog ne sme da bude prepun — najbolje je da se u njemu nalaze samo datoteke COMMAND.COM, AUTOEXEC.BAT, CONFIG.SYS i bitni direktorijumi. Kada otkucate DIR, čitav spisak programa treba da stane na jedan ekran. Drugi nivo direktorijuma treba takođe da bude slabo popunjen i pregledan, tek na treći nivo možete da nagurate veliki broj datoteka pošto četvrti nivo ne postoji.

Početnike pri kreiranju stabla kataloga muči jeden veliki problem: videli smo da integralni deo DOS-a čine eksterne komande BACKUP, FORMAT, PRINT... — ovih datoteka ima toliko da njihovim kopiranjem na hard disk *root* direktorijum postaje pretrpan. Rešenje je formiranje potkataloga koji ćete nazvati SYSTEM ili DOS i kopiranje eksternih komandi DOS-a u njega. Ostaje još da u datoteku AUTOEXEC.BAT ubacite liniju PATH C:\SYSTEM. Tačan smisao komande PATH je već opisan dok ćemo se datotekom AUTOEXEC.BAT uskoro baviti.

Naučimo još da navodimo puna imena datoteka. Datoteka AUTOEXEC.BAT u osnovnom katalogu zove se \AUTOEXEC.BAT (sećamo se da kosa crta na početku imena označava *root* direktorijum). Datoteka EDIT.EXE u direktorijumu JEZICI se zove \JEZICI\EDIT.EXE dok se datoteka PCDAT.TXT u direktorijumu ROTXT koji se sa svoje strane nalazi u direktorijumu TEKST zove \TEKST\ROTXT\PCDAT.TXT.

Vidimo da je navođenje imena direktorijuma prilično jednostavno — počnemo od *root*-a i navodimo sve kataloge kroz koje prolazimo razdvajajući ih obrnutim kosim crtama, na kraju navodimo ime same datoteke koje, jasno, obuhvata i tip.

Zadavanje direktorijuma u odnosu na *root* je jednostavno, ali zato, ako je stablo dublje, zahteva dosta kucanja. Zato iskusniji korisnici DOS-a često koriste relativno imenovanje: nalazimo se, na primer, u direktorijumu \TEKST\ROTXT i želimo da pređemo u direktorijum TEKST. Možemo da otkucamo CHDIR\TEKST ali i CHDIR.. — dve uzastopne tačke označavaju roditeljski katalog. Slično tome, ako se nalazimo u direktorijumu \TEKST\ROTXT a želimo da pređemo u TEKST\WPERF, možemo da otkucamo CHDIR JEZICI\WPERF ali i CHDIR ..\WPERF. Često se jasno pokazuje vrednost poslovice ko hoće veće, izgubi iz vreće — izrazimo se relativno, pročitamo poruku *Invalid directory* i onda ponovo kucamo celu naredbu!

Specijalne datoteke

Datoteka AUTOEXEC koja se nalazi u osnovnom *(root)* direktorijumu hard diska ili diskete sa koje „podižemo" sistem ima specijalan status: automatski se izvršava kadgod uključimo ili resetujemo PC. U ovu datoteku možemo da upišemo poruku kojom će nas računar pozdravljati i mnoge druge naredbe kojima prilagođavamo DOS sopstvenim potrebama i ukusima. Kako budete upoznavali PC tako će i vaša AUTOEXEC datoteka rasti i postajati komplikovanija.

Slika 5 prikazuje AUTOEXEC.BAT datoteku koju koristi autor ovoga teksta. Instalira se odgovarajući drajver za tastaturu, prelazi u direktorijum sa tekstom i, što je posebno važno, komandom PATH obezbeđuje izvršavanje eksternih DOS komandi. Korisnici PC-ja koji nemaju časovnik realnog vremena traba ovu datoteku da dopune komandama *date* i *time* koje zahtevaju kucanje datuma i vremena po svakom uključenju kompjutera.

slika 5.

```
ECHO OFF
PATH C:\SYSTEM, C:\JEZICI
MODE COM1: 1200 N 8 1
\UTILS\CED — F\UTILS\CED.DEF
KEYBUS
```

Dalje prilagođavanje DOS-a vašem ukusu postiže se uz pomoć datoteke CONFIG.SYS koja, poput AUTOEXEC.BAT, sadrži obične redove teksta, što znači da je formirate pomoću bilo kog tekst procesora, za nuždu možete da otkucate i COPY CON: CONFIG.SYS. Svaki red se sastoji od deklaracije i opcije, bitnim deklaracijama posvećujemo ostatak ovog poglavlja.

BUFFERS — podešava performanse diska. Vođeni činjenicom da je prvi IBM PC imao samo 64 kilobajta memorije, njegovi su konstruktori štedeli na svakom koraku pa je broj bafera za opštenje sa diskom održan na najmanjoj meri. Ukoliko uglavnom radite sa programima koji pristupaju disku (jedan od takvih je dBASE) povećanje broja bafera dovodi do dramatičnog objašnjenje performansi sistema — sortiranje i pretraživanje može da bude brže i celih 10 puta! zato u CONFIG.SYS uvrstite red poput BUFFERS=25.

COUNTRY — format prikazivanja datuma i vremena. MS DOS se prodaje širom sveta što znači da je u izvesnoj meri prilagođen korisnicima koji govore razne jezike. Ovo se prilagođenje svodi uglavnom na raspored tastera i način prikazivanja datuma. Dok ćemo prilagođavanje tastature upoznati u 35. poglavlju prikazivanje datuma podešavanje tastature upoznati u 35. poglavlju prikazivanje datuma podešavamo ga u okviru datoteke CONFIG.SYS. U našim se uslovima koristi COUNTRY=001 (SAD) ili COUNTRY=049 (Nemačke). Jugoslavije nema na spisku!

DEVICE — instalacija uređaja. Na PC se priključuju razni uređaji koje kontroliše odgovarajući softver. Taj softver prirodno može da se učita kada nam uređaj zatreba ali je ponekad korisno učiniti ga rezidentnim delom DOS-a i imati stalno na raspolaganju. Zato u CONFIG.SYS možete da ubacite deklaraciju DEVICE=*ime programa* kojom se unosi i aktivira program koji pokreće neki uređaj — taj uređaj može da bude egzotičan disk, miš, programator EPROM-a ali i video kartica odnosno RAM disk. Reč „uređaj" treba dakle shvatiti u širem smislu: to može da bude čitava kompjuterska mreža ali i deo hardvera samog PC-ja! Obično se sa DEVICE=MOUSE.SYS pokreće miš a sa DEVICE=VDISK.SYS 360 512 64 inicijalizuje RAM disk (vlasnici AT-a opremljeni megabajtom RAM-a poslednjoj deklaraciju dodaju /E i tako prostor za RAM disk rezervišu u *extended* memoriji koja i onako ne može da se koristi za nešto pametnije). Jednom instaliran, drajver uređaja je aktivan sve dok je računar uključen. Jedini (iole jednostavan) način da ga isključite jeste izbacivanje reda iz CONFIG.SYS i resetovanje sistema.

FILES — maksimalan broj otvorenih datoteka. Normalno startovan DOS se konfiguriše tako da u istom trenutku može da radi sa 8 datoteka od kojih su 5 rezervisane (tastatura, ekran i interfejsi su za DOS datoteke). Preostale tri datoteke su uglavnom dovoljne, ponekad ćete, međutim, doći do programa koji zahteva da u CONFIG.SYS ugradite deklaraciju FILES=20 koje neće izazvati neki veći gubitak memorije.

Slika 6 prikazuje datoteku CONFIG.SYS koju koristi autor ovog teksta. Povećan je broj drajvova i bafera, instaliran miš, drajveri ANSI i VDISK (RAM disk je u *extended* memoriji) i izabran američki format prikazivanja datuma. Datoteku možete da prenesete na svoj hard disk tako što ćete otkucati CD (tako prelazite u osnovni *root* direktorijum) a zatim COPY CON: CONFIG.SYS. Pošto otkucate sve redove sa slike 6.6 pritisnite CTRL I Z.

slika 6

```
BREAK ON
BUFFERS=25
COUNTRY=001
DEVICE=ANSI.SYS
DEVICE=MOUSE.SYS
DEVICE=VDISK.SYS 360 512 64\E
FILES=20
LASTDRIVE=D
```

Komercijalni softver

Biblioteka komercijalnih programa za IBM PC sadrži nekoliko desetina hiljada naslova. U okviru ovog teksta pokušaćemo da opišemo neke osnovne primene PC-ja i nabrojimo važnije komercijalne programe koji te primene podržavaju. Kupovina računara i nabavka programa, međutim, predstavlja samo prvu stepenicu u rešavanju problema — za savladavanje ostalih stepenica morate se neoružati značajnom dozom upornosti

Pre svih komercijalnih programa treba izabrati sam operativni sistem. To će svakako biti MS DOS, ali koja verzija MS DOS-a? Po Jugoslaviji kruže verzije 2.00, 2.11, 3.00, 3.10, 3.20 i 3.30, koje se, i pored različitih brojeva, u suštini mnogo ne razlikuju. U ovom je trenutku najbolje izabrati DOS 3.30, ali ni izbor DOS-a 3.10 nije loš — radi se o verziji koju koristi najveći broj YU PC-jevaca. Pošto se odlučite za neku od verzija, treba da ostanete pri toj odluci — paralelno korišćenje raznih verzija DOS-a posle izvesnog vremena postaje prava mora!

Programiranje

Nisu daleko amaterski dani kada se smatralo da planinski vrhovi služe za osvajanje, a računari za programiranje. Obzirom da je koncepcija uvođenja informatike u škola zasnovana na programiranju, za nastavnike je naročito značajno upoznavanje programskih jezika koje jedan PC podržava.

Obično se počinje od bejzika: ukoliko u ROM vašeg PC klona nije ugrađen odgovarajući interpretator, moraćete da nabavite disketu sa GWBASIC-om. GWBASIC je veoma kompletna varijante najmasovnijeg kompjuterskog jezika koja podržava brojne tipove podataka, kontrolnu strukturu WHILE ... WEND, procedure sa prenosom parametara, rad sa datotekama, pristup grafici i tome slično. Brzina izvršavanja nije baš impresivna, ali je na raspolaganju relativno star Microsoftov bejzik kompajler koji značajno ubrzava programe čiji je razvoj završen prevodeći ih na mašinski jezik. Druga ozbiljna mana GWBASIC-a je zahtev da program i podaci ne zauzimaju više od 64 kilobajta RAM-a.

Sa GWBASIC-om se takmiče mnogi drugi bejzik interpretatori i kompajleri čije su karakteristike uglavnom znatno bolje, ali koji se, usled inercije vlasnika PC-ja, slabo koriste. Pomenućemo sve popularniji QuickBASIC i Borlandov Turbo bejzik koji zapravo predstavlja kompajler koji izvorni program čudesno brzo prevodi u mašinski — skoro kao da radite sa interpretatorom! Popularnosti Turbo bejzika u našim krajevima značajno smeta jedino njegova relativna nekompatibilnost sa Herkules karticom.

Već smo se, dakle, susreli sa Microsoftovim i Borlandovim kompajlerima — utakmica ove dve firme neprekidno traje, tako da su na tržištu dve paralelne i oštro suprotstavljene familije jezika. Borland je zadobio brojne poene lansirajući zaista izvanredan paskal kompajler nazvan, neizvesno, Turbo Pascal — ovaj program je naročito značajan za škole, pošto mnogi autoriteti smatraju da prvi programski jezik koji deca uče treba da bude paskal ili modula 2, a nikako bejzik. Najveća prednost Turbo paskala je izuzetno komforan razvoj programa — kompajler je integrisan sa editorom, tako da programer može da ispravi greška čim ih računar pronađe. Samo prevođenje je, uz to, toliko brzo da ga, osim kod veoma dugačkih programa, i ne primećujemo. Ranije verzije Turbo paskala nisu prevazilazile limit od 64 kilobajta za program i podatke, ali je na verziji 4.0, koja je ugledala svet početkom ove godine, taj problem otklonjen. Microsoftov paskal može da se nabavi, ali nam nije poznato da ga neko koristi.

Programski jezik C još nije stekao neku posebnu ulogu u školama, ali ga profesionalni programeri intenzivno koriste pri razvoju većih paketa. Korisnici PC-ja mogu da izaberu Microsoftov, Borlandov, Mark Williams ili (ne baš preporučljiv) Lattice C. Microsoftova varijanta je prilagođena raznim verzijama Intelovih mikroprocesora (čak i novom 80386) i dopunjena izvanrednim dibagerom *CodeView* o koma ćemo tek govoriti. Turbo C je uglavnom usmeren prema brzini prevođenja, dok Mark Williams C predstavlja zanimljivu kombinaciju kompajlera i linkera koja je raspoloživa na raznim ne-PC kompatibilnim računarima. Verujemo da je u ovom trenutku Microsoftov C najbolji izbor, jer se programiranje na C-u često kombinuje sa programiranjem u asembleru, pa je zgodno koristiti bateriju prevodilac — asembler — dibager istog proizvođača.

Najstariji kompjuterski jezik fortran zastupljen je sa nekoliko kompajlera: prednost se uglavnom daje Microsoftovom MS fortranu i Professional Fortran-u. Skup podržanih instrukcija je prilično sličan fortranu 77 (ponešto je ipak izostavljeno), ali rad nije naročito komforan jer je prevođenje i povezivanje relativno sporo. Fortran kompajleru je objektivno potreban aritmetički koprocesor: pa će neke verzije prevodioca čak i odbiti da rade bez njega, druge će raditi, ali će i prevedeni program biti prilično spor.

Od ostalih jezika vredi pomenuti Turbo Prolog, par modula 2, kobol, PL/1 i razne „hardverski orijentisane" jezika kao što je PL/M. Svi pomenuti kompajleri (osim Turbo prologa) ne mogu da se pohvale prevelikim komforom pri radu i često predstavljaju više demonstraciju nekog jezika nego upotrebljivu alatku.

I na kraju — asembler. PC se, istini za volju, retko programira na asembleru (za ovakvim programiranjem objektivno nema mnogo potrebe), ali u svojoj biblioteci svakako treba imati neki asembler koji će zatrebati pri integrisanju raznih programskih paketa. Kažemo *neki asembler*, iako tu konkurencije praktično i nema — pitanje je samo koju ćete verziju Microsoftovog asemblera uspeti da nabavite. Kod nas je vrlo česta neprihvatljivo spora verzija 2.0, ali se ponegde može naći i Macro Assembler 4.0; verzija 5.0, koliko nam je poznato, još nije stigla do Jugoslavije. Posebnu pažnju izaziva dibager *CodeView* koji pojednostavljuje pronalaženje i ispravljanje grešaka u mašinskim i, što je posebno interesantno, prevedenim programima koji su pisani na nekom višem jeziku — kažemo da se radi o *source level* dibageru — dibageru na nivou izvornog koda.

IBM PC je, sve u svemu, mašina koja može izvanredno da se primeni za obrazovanje programera — na raspolaganju su gotovo svi zamislivi kompajleri i interpretatori. Kvalitet mnogih od njih, jasno, nije na potrebnoj visini, ali će čak i najprobirljiviji autori programa svakako pronaći razvojna alatke koje će ih zadovoljiti.

Obrada teksta

Obrada teksta je ubedljivo najpopularnija primena personalnih računara — malobrojni su doduše srećnici obdareni sposobnošću i prilikom da pišu romane, pripovetke i mirovne ugovore, ali je zato krug pisaca pisama, zapisnika, tužbi, žalbi i sličnih administrativnih zavrzlama veoma širok. Pomoć koju računar može da pruži pri obradi teksta je za početnika gotovo nezamisliva — još se nismo sreli sa nekim ko bi se, upoznavši tekst procesor, vratio pisaćoj mašini! IBM PC je potencijalno izvanredna mašina za obradu teksta, ali izbor tekst procesora nije baš jednostavan.

Jugoslavijom kruži ogromna količina PC softvera čiji je kvalitet često veoma sumnjiv — nedostatak dokumentacije je činjenica na koju su se svi privikli, ali mnogi korisnici nakon programa tek posle dugog vremena primete da su neka komponenta programskog paketa uništena nestručnim kopiranjem, čime su mogućnosti programa naprosto prepolovljene. Pokušaćemo da nabrojimo nekoliko programa za obradu teksta koji su postigli određenu popularnost na našim meridijanima i čija je upotreba uslovno rečeno bezbedna: iako se može naći korektna verzija i relativno opsežna dokumentacija koja je ponekad čak pisana i na našem jeziku!

PC u „Računarima"

Asemblersko programiranje: Tiha voda brega roni (17/26).
Bejzik programiranje: Bilo kuda — bejzik svuda (22/46).
Biblioteka programa: Send (17/37). Reci DOS u NE! (36/51).
Grafika i animacija: Lična fabrika snova (animacija) (31/16).
Interfejsi: Dnevnik jedne veze (29/24). Sa amstrada na PC (35/38).
Komercijalni softver: Jezički procesori u 16 lekcija (12/46). Misli kao što pišeš (13/39). PC za svaki dan (19/14). Istinski bejzik (22/39). Borlandov turbo paskal (23/56). Turbo bejzik (26/28). Prave alatka za Prave Programere (28/54). 77 PC programa (29/29). Emulacija ili imitacija (29/53). Tekuće verzije kompajlera (30/6). Isterivači bubica (dibagera) (33/22). Četvorac sa programerom (QuickBasic) (35/32). Ne daj bubi da te gricka (CodeView) (36/32).
Nabavka PC-ja: Hakerski vodič Minhena (16/12). PC kombinacija snova (17/18). Tata kupi mi PC (32/14).
Novi PC računari: Neka bude IBM (14/17). Šećerni PC (20/9). Gospoda s Tajvana (21/30). PC na mlazni pogon (Compaq 386) (22/8). Tandi se vraća kući (23/8). Teška artiljerija (24/8). Klon sa klonom izbija (Atari PC (25/16). Amiga ljubavi moje (26/17). Rapsodija u plavom (27/9). Plavi petuljak (PS/2 model 30) (29/10). I to se zove PC (Atari PC) (30/8). Popravni iz likovnog (PC 1640) (30/12). Napred plavi (PS/2 model 50) (32/10). Ne daj se generacijo (PS/2 80) (35/10). Originali i klonovi (96 klonova) (36/21).
Obrada teksta: Lakše se piše (19/17). Treća sreća (Word 3) (20/16). Štamparija na stolu (28/51). Sam svoj izdavač (34/26). Stotinu zato za stono (36/18). Unos i priprema teksta (36/17).
Operativni sistem: Veliki i mali (PC — C64) (14/22). PC bukvar (18/41). Carevo novo ruho (GEM) (26/57). Ne naginji se kroz prozor (27/58). Sad se vidi, sad se zna (OS/2) (30/14). MS DOS za početnika (31/16).
Performanse PC-ja: Sve IBM-ove funkcije (23/60). Bagovi u plavom (24/19). Sprinteri na duge staze (80386 vs 68020) (32/12).
Periferijska oprema: Računar koji čita (Digitizer) (31/14).
Pratеći čipovi: Forth u čipu (23/22). Matematičar gvozdenog kova (arit. kopr.) (29/20). Frizirana sedmica (80287) (35/20).
Prikaz knjige i časopisa: ABC PC (30/24). IBM PC — Uvod u rad DOS Basic (32/26). IBM PC/AT/XT u 25 lekcija (32/27). Aplikativni programi za PC i Apple 2 (33/21).
Programski jezici: Turbo je nešto drugo (Turbo Pascal) (20/16). Borlandovo visoko C (34/20). Objektno programiranje (36/28).
Proširenja PC-ja: Prava slika za prave programera (PC) (19/18). Bejzik protiv Herkulesa (21/38). Sve PC kartice (27/22). Ko je oodjio Herkulesa? (29/12). Steti i gledeti (Wyse 700) (35/15).
Samogradnja: Ukroćeni Herkules (34/22).
Struktura ROM-a i operativnog sistema: PC usluga na niskom nivou (24/16). Kako startovati program (MS DOS) (25/32). Kao po meri (nove naredbe) (26/36). Sve MS DOS funkcija (26/27). Ostajete ovde (rezidentni programi) (32/52).
Tehnika programiranja: Podešavanje tastatura (25/34). Sedam prljavih (paskal) trikova (33/28). Ličnu kartu, molim (prepoznavanje mikroprocesora) (35/44).

WordStar je najstariji i najslavniji program za obradu teksta — mnogi profesionalci ga poznaju, ali ga sve manje ljudi koristi. *WordStar* je, naime, neprijatan za početnike, od kojih zahteva pamćenje složenih kontrolnih sekvenci i unekoliko programerski pristup obradi teksta. Druga mana *WordStar*-a je spor rad — program neprekidno komunicira sa spoljnom memorijom, što predstavlja neprijatno usporanje čak i za vlasnika hard diskova. *WordStar* ćete, dakle, upoznati samo ako tekst treba da obrađujete na mnogim računarima iz različitih generacija — sva je prilika da je na svakom od njih instalirana po neka verzija *WordStar*-a.

Microsoft Word je daleko moćniji program, prilagođen početnicima ali je ipak interesantan i za iskusnije korisnike — komunikacija sa računarom svodi se na izbor stavki iz raznih menija i šetnju mišem po stolu. Starije verzije *Word*-a (po Jugoslaviji kruži verzija 3.0, a nedavno je izašao i *Word 4.0*) bila su prilično spora (pogotovu za korisnike koji još nisu nabavili hard disk), ali *Word*-u 4.0 stvarno nema šta da se zameri. Ovaj je program interesantan i za korisnike koji se bave takozvanim „stonim izdavaštvom", tj. kompjuterskom pripremom rukopisa za štampu — *Word* je pogodan za precizno određivanje oblika strane, razmaka između redova, raznih margina i tome slično. Prave je šteta što je rad sa tabulatorima prilično iskomplikovan.

WordPerfect (tekuća verzija je 4.2) je tekst procesor namenjen unekoliko naprednijim korisnicima — komunikacija sa računarom se svodi na pritiskanje funkcijskih tastera, ali računar pri tome aktivno pomaže korisniku upozoravajući ga na mogućnosti koje mu stoje na raspolaganju i ispisujući menije kada su oni stvarno potrebni. Izrazito dobre strane *WordPerfect*-a su brz rad, racionalno korišćenje memorije i diska i moćan prateći program za kontrolu spelovanja, a nedostatak (u odnosu na *Word)* manja fleksibilnost pri konačnom formatiranju teksta.

PC Writer i *IBM Writing Assistant* koriste vlasnici PC-ja koji, obzirom da se uglavnom bave pismima i kraćim tekstovima, još nisu nabavili hard disk — programi sasvim pristojno rade i sa disk jedinicama. Žrtva kompaktnosti su karakteristika — *PC Writer*-u i *IBM Writing Assistant*-u nedostaju mnoge mogućnosti *WordPerfect*-a ili *Word*-a.

Chi Writer je „naučni" tekst procesor koji je dizajniran tako da olakšava unošenje matematičkih formula. Punu snagu postiže tek ako je vaš PC povezan sa laserskim štampačem i dopunjen odgovarajućim drajverima.

U poslednje vreme sve više pristalica stiče već pomenuto stono izdavaštvo — računar opremljen laserskim štampačem uspešno zamenjuje foto slog i tako bitno ubrzava i pojeftinjuje pripremu rukopisa za štampu. Koristi se uglavnom program *Ventura* čiji opis izlazi iz okvira ovako opšteg teksta.

Propisno konfigurisan IBM PC je, sve u svemu, gotovo idealna mašina za obradu teksta. Izbor programa uglavnom se svodi na dve alternative, odnosno na dve koncepcije — *Word* i *WordPerfect*-a; odluka zavisi uglavnom od vašeg predznanja i ukusa.

Baze podataka

Posle obrade teksta, vlasnici kućnih računara najčešće rade sa bazama podataka. Programi ovoga tipa nam pomažu da u memoriji kompjutera i na disku držimo velike skupove raznih podataka koji se mogu brzo i efikasno menjati, pretraživati i štampati. Ukoliko je baza podataka integrisana sa tekst procesorom, koristićemo je za pisanje službenih pisama i štampanje spiskova članova nekog kluba ili pretplatnika na neku knjigu i tako bitno pojednostaviti i pojeftiniti organizaciju posla.

Pojava personalnih računara približila je baze podataka širokom krugu zainteresovanih, pa su i komercijalni programi koji obezbeđuju ovakve primene sve bolje prolazili na tržištu. To su u početku bile „igračke" tipa telefonskog imenika, podsetnika ili programa koji sređuje manju biblioteku knjiga, a onda se pojavio dBASE, program koji je, istina, nastao na CP/M-u, ali koji je veoma zaslužan za današnji status IBM PC-ja. Program *dBASE III Plus*, uzgred budi rečeno, trenutno nije ni najbolja, ni najjeftinija, ni za upotrebu najjednostavnija baza podataka prilagođena PC-ju: on ipak predstavlja toliko rasprostranjen industrijski standard da njegovo poznavanje praktično predstavlja deo osnovne škole računarstva. Ukoliko, dakle, želite da formirate i održavate bazu podataka na PC-ju, treba da upoznate dBASE.

Unakrsna izračunavanja

Dok je obradu teksta i baze podataka lako objasniti svakom laiku, potreba za unakrsnim izračunavanjima nije baš očigledna. Da pokušamo ovako: potpisali ste niz brojeva jedan ispod drugog, povukli crtu i onda te brojeve, uz razne komentare tipa *tri pišem, jedan pamtim*, sabrali. Za ovakvo nešto teško da je potreban IBM PC. PC, međutim, dolazi do izražaja kada neki od sabiraka treba promeniti: pozicionirate kurzor u odgovarajuće polje, otkucate novu vrednost i zbir se automatski i trenutno ažurira. Kada se zbirovi pretvore u složene izraze, a niz brojeva preraste u dvodimenzionalnu tabelu, program za unakrsna izračunavanja dolazi do punog izražaja — sa pravom se smatra da ovakav program približava brojne primene računara ljudima koji uopšte nisu upoznati sa programiranjem, pa čak ni sa upotrebom kompjutera.

Vodeći programski paket za unakrsna izračunavanja je Lotus 1—2—3 — ovaj program, poput dBASE-a, verovatno nije najbolji na tržištu, ali njegova tradicija garantuje da ćete uspešno rešiti svaki rešiv problem i da će vam pomoć u nevolji pružiti brojna literatura (na žalost, još ne na našem jeziku!) i mnogi iskusniji korisnici. Dodatak uz unakrsna izračunavanja je poslovna grafika — rezultate možete da prikažete kao dijagrame, histograme, „pite" i slične strukture.

Crtanje i projektovanje

„Četvrti musketar" (uz *Word Star*, *dBASE* i *Lotus*) PC-jeve biblioteke je *AutoCAD*, program koji olakšava računarsko crtanje i projektovanje. Pod crtanjem ovde ne podrazumevamo modernu ili klasičnu umetnost, već tehničke crteže i planove — *AutoCAD* je dizajniran tako da eliminiše tehnički najneprijatniji deo posla (višestruko ponavljanje istih operacija i identičnu obradu raznih slika) i omogući korisniku da se usredsredi na kreativan deo projektovanja. Treba ipak znati da *AutoCAD* zahteva solidno konfigurisan PC ili, još bolje, AT — aritmetički koprocesor i hard disk od dvadesetak megabajta su apsolutni minimum, ali se pravi rezultati dobijaju tek primenom kvalitetnog plotera ili laserskog štampača.

Domaća PC literatura

Iako se domaća računarska publicistika do skora bavila isključivo „spektrumom", „komodorom 64" i „amstradom 464/6128", smatrajući da vlasnici ovih kompjutera predstavljaju najšire potencijalno tržište, popularnost IBM PC-ja je učinila svoje — tokom 1987. na tržištu se pojavilo nekoliko interesantnih naslova.

Prva knjiga, kako hronološki tako i po pristupu, je „ABC PC" Zorana Životića u izdanju Zavoda za udžbenike i nastavna sredstva (Beograd) i Zveze organizacij za tehničko kulturo Slovenije (Ljubljana). Knjiga je koncipirana slično ovom tekstu: osnovne informacije o nabavci, sklapanju, upotrebi i primeni IBM PC-ja prilagođena apsolutnom početniku. Samo se po sebi razume da je prostor u knjizi obezbedio daleko detaljniju prezentaciju podataka.

„IBM PC — Uvod u rad, DOS, BASIC" Stevana Milinkovića, Vladimira Jankovića i Dragana Taneskoskog u (privatnom) izdanju beogradske „Mikro knjige" je detaljan DOS priručnik koji opisuje upotrebu PC-jevog operativnog sistema i GWBASIC interpretatora. Knjiga je veoma pogodna za početnike i iskusnije korisnike koji se ne snalaze sa engleskim jezikom, pa ne mogu da iskoriste blagodeti stranih uputstava za upotrebu PC DOS-a.

„IBM PC/AT/XT u 25 lekcija" Vojislava Mišića u izdanju „Tehničke knjige" (Beograd) je knjiga koja će vas postupno upoznati sa hardverom i softverom IBM PC-ja — prve lekcije namenjene su apsolutnim početnicima ali se, približavanjem broju 25, stvari značajno usložnjavaju. „IBM PC u 25 lekcija" je, uz obilje informacija, štivo veoma zanimljivo za čitanje.

Početak 1988. doneo nam je PC literaturu koja nije namenjena programerima — „Tehnička knjiga" je izdala tri priručnika za *korisnike* PC-ja. To su, hronološkim redom, „Aplikacioni programi za personalne računare" Dragana Pantića, „Obrada teksta na računaru" autora ovog teksta i dBASE priručnik koji će se pojaviti u prodaji otprilike kada i ovo specijalno izdanje. Prva knjiga se ukratko bavi obradom teksta, bazama podataka i unakrsnim izračunavanjima na IBM PC-ju i „eplu II", druga obradom teksta na IBM PC-ju, „komodoru 64/128" i „amstradu 6128" (detaljno je opisan već pomenuti program *WordPerfect)*, a treće najpopularnijim programom za rad sa bazama podataka dBASE III Plus. Verujemo da će i sledećih meseci izdavači nastaviti da nas snabdevaju PC literaturom za programere i početnike.

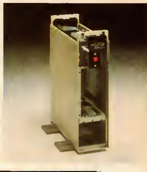
Treba, naravno, pomenuti i doprinos kompjuterskih časopisa koji se već nekoliko godina intenzivno bave PC-jem. Ostatak ovog poglavlja posvećujemo svojevrsnom indeksu tekstova o IBM PC-ju koji su objavljivani u časopisu „Računari" — tekstovi su sortirani po temama i u okviru svake teme po broju časopisa i strani: 17/26 znači da je tekst na 26. strani „Računara 17". Posebno vredi pomenuti naše umetke (obim oko 100 kucanih ili 32 štampane strane): *MS DOS za početnike* („Računari 31"), *77 PC programa* („Računari 29"), *dBASE III Plus* („Računari 37"), *Sve MS DOS funkcije* („Računari 26"), *Sa bejzika na paskal* („Računari 24") i *Sa bejzika na C* („Računari 27") — prvi opisuje upotrebu MS DOS-a, drugi predstavlja poznate komercijalne programe za PC-je, treći se bavi dBASE-om III, četvrti je namenjen programerima koji uz pomoć usluga operativnog sistema žele da skrate svoje programe, a preostala dva predstavljaju popularne škole paskala i C-a prilagođene, pored ostalog, i PC-ju. Za ovu godinu pripremili smo seriju umetaka sa aplikativnim softverom: pored programa dBASE II plus biće predstavljeni i programi Lotus 1—2—3 („Računari 36"), Ventura, Word, Word Perfect, Framework i, verovatno, AutoCAD.



Honeywell

RI MARKETING
11000 BEOGRAD Masarikova 5/18
tel. 641-555 686-047
telex 11820





X-SUPERTEAM®

Koncipiran kao mali računar opšte namene — baziran na moćnim 32-bitnim mikroprocesorima i veoma traženom operativnom sistemu UNIX-X SUPERTEAM (modeli X20 i X40) izvanredan je, između ostalih aplikacija, i kao nezamenljivo sredstvo u obrazovnom procesu (škole i fakulteti)

Programi za rad X-SUPERTEAM/UNIX

Da bismo bolje objasnili fenomen Unixa moramo pre svega objasniti pojam standardnog operativnog sistema koji se pojavio u poslednje vreme. Naime operativni sistemi su kao najkompleksniji deo softvera uvek bili projektovani za određeni tip računara određenog proizvođača, uglavnom su napisani u asembleru za određeni hardver (ima operativnih sistema koji su napisani u višem jeziku međutim oni pozivaju toliko potprograma u asembleru da time gube prednost). Evidentno je da je prenos takvog operativnog sistema na drugi hardver ukoliko bi ga proizvođač uopšte ustupio čista utopija. Svaki od tih sistema ima mnogo specifičnosti tako da su konverzije aplikacija sa jednog na drugi operativni sistem skup i dugotrajan proces. Jasno je da korisnik takvog sistema može da koristi samo aplikacije razvijane baš za taj sistem a čija je raspoloživost ograničena na određene oblasti i namene.

Pojavom prvih jeftinih i široko raspoloživih mikroprocesora počele su da se dešavaju čudne stvari. Na tržištu se pojavila velika količina jeftinog hardvera. Proizvođači nisu imali računa da zbog niskih prodajnih cena sami razvijaju softver za takve sisteme pa su se u to što organizovano što spontano upustile nezavisne softverske kuće. Pošto se softver za jeftin hardver ne može prodati skupo te kuće su morale da žive od prometa tj. implementiraju isti softver na sistemima raznih proizvođača. Posle izvesnog vremena postao je dominantan onaj operativni sistem koji je u prvom koraku razvoja računara na bazi mikroprocesora bio prihvaćen od većeg dela proizvođača. Za njega je bilo raspoloživo najviše aplikativnog softvera, zbog toga ga je prihvatalo sve više korisnika, zbog toga je sve više i više kuća radilo aplikativni softver za njega, i tako se magični krug zatvorio. Prvi primer uparivanja standardnog mikroprocesora i standardnog operativnog sistema su Zilog Z80 i CP/M. Isti proces se još više eskalirao sa MS/DOS-om i Intelom 8086.

Ova dva slučaja nisu imala većih implikacija na postojeću ponudu na tržištu računara. Radilo se o potpuno novom proizvodu, personalnim računarima male moći koji su bili namenjeni novom segmentu tržišta i dopunjavali postojeću ponudu. Pojavom 16-bitnih i 32-bitnih mikroprocesora i jeftine periferije stvoreni su sistemi koji svojim performansama mogu da konkurišu mini i manjim srednjim računarima. Zbog svojih osobina i raspoloživosti u pravom trenutku Unix je za ove sisteme postao standardni operativni sistem — danas postoje kvalifikovane verzije za sve rasprostranjenije mikroprocesore. Razlika između Unixa i dva već pomenuta standardna operativna sistema je u tome što Unix konkuriše „sopstvenim" operativnim sistemima na već etabliranom tržištu mini i malih računara opšte namene. Taj pritisak se uočava kroz to što većina velikih proizvođača implementira Unix na sopstvenim sistemima. Na taj način Unix postaje „standardan" ne samo za određenu klasu mikroprocesora već i za klasu računara određene snage. Taj proces pospešuje relativno jednostavno implementiranje Unixa napisanog u C jeziku na bilo koji sistem. (Prema nekim podacima za prenos Unixa na novi hardver je potrebno manje čovek/meseci angažovanja nego za pripremu nove revizije „sopstvenog" operativnog sistema).

Druga činjenica koja bitno utiče na eskalaciju Unixa je demokratizacija korišćenja računara prouzrokovana uglavnom zahvaljujući pojavi personalnih računara. Danas mnogi imaju na svom radnom mestu ili kod kuće personalni računar, navikli su da koriste softverske alatke i pakete koje pruža MS/DOS. Normalno je da mnogima PC postaje tesan. Rešenje je saradnja sa nekim jačim računarom. Emulacija terminala nije zadovoljavajuće rešenje jer su softveri i formati datoteka na dva računara nekompatibilni. Potrebna je neka vrsta integracije PC-host računar tj. iste softverske alatke i kompatibilne datoteke. Upravo je tu velika prednost Unixa, većina softverskih paketa sa personalnih računara je ili će uskoro biti raspoloživa i pod Unixom.

Unix se i dalje razvija. Jedan od glavnih pravaca je standardizacija komunikacija. Verovatno ste već primetili da je o komunikacijama do sada bilo malo pomena, uglavnom zbog toga što je Unix do skora bio namenjen timovima koji su radili na razvoju softvera i imali malo potrebe da preko računara komuniciraju sa spoljnim svetom. Zbog toga je svaki proizvođač obezbeđivao mogućnost komunikacija po sopstvenom nahođenju, uglavnom komunikacije sa host sistemima i uključenje u javne mreže za prenos podataka. Danas se radi na standardizaciji komuniciranja Unixa u dva smera. Jedan je integracija inteligentnih radnih stanica (PC) preko lokalnih mreža tipa Ethernet, drugi smer je naviše ka velikim host sistemima i javnim mrežama. Osim na komunikacijama radi se i na značajnom poboljšanju podrške poslovnim primenama, aspektu kojem do početka



komercijalizacije Unixa nije poklanjano mnogo pažnje

Ako rezimiramo, korisnik Unixa ima nekoliko značajnih prednosti

— zaštitu investicija u sopstveni razvoj softvera (eventualna promena računara ne implicira konverziju aplikacija)

— veliki (s tendencijom porasta) broj gotovih softverskih paketa na tržištu

— operativni sistem koji se i dalje intenzivno razvija

— zadovoljstvo što koristi Unix

X-SUPERTEAM

Honeywell nije pridošlica u svet UNIX-a. Naprotiv, Ritchie i Thompson su bili deo tima iz Bell Labs koji je učestvovao u razvoju MULTICS sistema na računaru GE 645. MULTICS je preteča savremenih Honeywell-ovih operativnih sistema serije GCOS a GE 645 je preteča Honeywell-ovih velikih računara DPS 8/88/90. Dobar deo ideja iz projekta MULTICS Ritchie i Thompson su iskoristili za koncepciju UNIX-a. Linija računara X-SUPERTEAM znači povratak u Honeywell ideja i koncepata koji su u međuvremenu obogaćeni novim dostignućima na polju računarske arhitekture i tehnologije i programske opreme.

Zašto UNIX? Današnji trend računarstva je standardizacija zbog integracija, zbog kooperacije, zbog zaštite velikih investicija koje savremeni software zahteva. UNIX je zbog svoje fleksibilnosti, svoje otvorenosti, svoje savremenosti postao de facto standard operativnih sistema. Ulazeći u svet UNIX-a korisnik ima na raspolaganju gotovo neograničen broj software-skih oruđa i aplikativnih paketa, može sopstvene aplikacije prenositi na druge računare bez ikakvih modifikacija, jednom rečju svuda se oseća kao kod kuće. Honeywell je, ulazeći u svet UNIX-a i industrijskih standarda, tu otvorenost obogatio i dizajnom svojih 32-bitnih supermikro računara zasnivajući ih na standardnim komponentama i arhitekturi koja dozvoljava obogaćivanje elementima nevezanim za određenog proizvođača.

X-SUPERTEAM je koncipiran kao mali računar opšte namene baziran na moćnim 32-bitnim mikroprocesorima. Arhitekturu karakteriše dualna magistrala, jedna za vezu procesor — memorija a druga za vezu memorija — periferija. Budući da je magistrala za periferiju standardna (VME bus) to je moguće priključiti sve kontrolere raspoložive na tržištu. Pridružena "cache" memorija znatno povećava performansu procesora.

Standardnu periferiju predstavljaju fiksni diskovi (72/143 MB formatirano), fleksi disk (720 KB/1.2 MB) i kasetni streamer (45/60 MB). Na veće modele je moguće priključiti standardne GCR/PE jedinice magnetnih traka (1600/6250 BPI). Ponudu dopunjuje bogat izbor Honeywell-ovih štampača i terminala. Standardno se priključuju kontroleri za ETHERNET LAN, X.25 i SNA. Operativni sistem je UNIPLUS, derivat UNIX-a V 2.0 razvijen sa AT T. Raspoloživi su svi važniji programski jezici — COBOL, FORTRAN, Pascal, BASIC, C jezik. Za struktuiranja podataka se može koristiti C-ISAM ili UNIFY — software za upravljanja bazama podataka. Za automatizaciju uredskog poslovanja su raspoloživa dva rešenja, UNIPLEX II+ i ALIS koji trenutno predstavlja vrh ponude u ovoj oblasti na svetskom tržištu.

Software i hardware dozvoljavaju maksimalnu integraciju u geografski distribuiranu mrežu. Moguća je interaktivna komunikacija i udaljeni unos poslova IBM hostu, bilo kroz klasične 3270/2780/3780 protokole, bilo kroz emulaciju SNA PU T2 (3274/3777). Komuniciranje sa Honeywell hostom je omogućeno emulacijom VIP protokola. X.25 protokol omogućuje uključenje u javne mreže za prenos podataka. Lokalne mreže su podržane hardware-om i software-om koji realizuju ETHERNET standard. U takvu mrežu je moguće uključiti više X-SUPERTEAM sistema, pa je moguće vršiti prenos datoteka, udaljenu štampu, pristup aplikacijama na drugom sistemu i elektronsku poštu. Posebnu prednost ovakve mreže predstavlja mogućnost integracije ličnih računara (PC) gde X-SUPERTEAM između ostalog funkcioniše i kao file server za MS DOS.

Tabelarni pregled karakteristika X-SUPERTEAM-a po modelima

X-SUPERTEAM	X20	X40
PROCESOR	MC68020	2xMC68020
CACHE	OPT	STD
MEMORIJA (MB)	2—10	4—20
DISKOVI (MB)	435	870
ŠTAMPAČI	4	8
LINIJE	32	64

tim011



ŠKOLSKI RAČUNAR



ŠKOLSKI RAČUNAR TIM 011

TIM 011 je najnoviji model iz familije TIM računara namenjen opštem i profesionalnom obrazovanju u oblasti informatike i računarstva kao i unapređenju nastave.

TIM 011 je pogodan za efikasno vođenje školske administracije kao i za povezivanje sa centrima koji se bave obrazovnom problematikom (biblioteke, univerziteti itd.).

TIM 011 - do 10 računara povezanih sa profesionalnim školskim računarom TIM 020 kompatibilnim sa PC XT, predstavljaju snažnu laboratoriju za informatiku i računarstvo

TIM 011 je usvojen u Beogradu kao standard za osnovno i usmereno obrazovanje

PROGRAMSKA OPREMA:

- ***Operativni sistem:***
  Usavršen i proširen CP/M kompatibilan, disk operativni sistem
- ***Programski jezici:***
  BASIC INTERPRETATOR
  BASIC PREVODILAC
  FORTRAN
  COBOL
  C
  PASCAL
  MODULA 2
  PROLOG
  LOGO
- ***Veliki broj uslužnih programa,***
  procesor teksta,
  baza podataka,
  kalkulacije,
  školski aplikativni
  programi

Institut MIHAILO PUPIN